8-2017

Estimating Accuracy of Personal Identifiable Information in Integrated Data Systems

Amani "Mohammad Jum'h" Amin Shatnawi
Utah State University

ESTIMATING ACCURACY OF PERSONAL IDENTIFIABLE INFORMATION IN INTEGRATED DATA SYSTEMS

by

Amani “Mohammad Jum’h” Amin Shatnawi

A dissertation submitted in partial fulfillment
of the requirements for the degree

of

DOCTOR OF PHILOSOPHY

in

Computer Science

Approved:

_____ Stephen Clyde, Ph.D. Major Professor	_____ Xiaojun Qi, Ph.D. Committee Member
_____ Curtis Dyreson, Ph.D. Committee Member	_____ Nicholas Flann, Ph.D. Committee Member
_____ Bedri Cetiner, Ph.D. Committee Member	_____ Mark R. McLellan, Ph.D. Vice President for Research and Dean of the School of Graduate Studies

UTAH STATE UNIVERSITY
Logan, Utah

2017

ABSTRACT

Estimating Accuracy of Personal Identifiable Information in Integrated Data Systems

by

Amani Shatnawi, Doctor of Philosophy

Utah State University, 2017

Major Professor: Stephen W. Clyde, Ph.D.
Department: Computer Science

Without a valid assessment of accuracy there is a risk of data users coming to incorrect conclusions or making bad decision based on inaccurate data. This dissertation proposes a theoretical method for developing data-accuracy metrics specific for any given person-centric integrated system and how a data analyst can use these metrics to estimate the overall accuracy of person-centric data.

Estimating the accuracy of Personal Identifiable Information (PII) creates a corresponding need to model and formalize PII for both the real-world and electronic data, in a way that supports rigorous reasoning relative to real-world facts, expert opinions, and aggregate knowledge. This research provides such a foundation by introducing a temporal first-order logic language (FOL), called Person Data First-order Logic (PDFOL). With its syntax and semantics formalized, PDFOL provides a mechanism for expressing data-accuracy metrics, computing measurements using these metrics on person-centric databases, and comparing those measurements with expected values from real-world populations. Specifically, it enables data analysts to model person attributes and inter-person relations from real-world population or database representations of such, as well as real-world facts, expert opinions, and aggregate knowledge. PDFOL builds on existing first-order logics with the addition of temporal predicated based on time intervals, aggregate functions, and tuple-set comparison operators. It adapts and extends the traditional aggregate functions in three ways: a) allowing any arbitrary number free variables in function statement, b) adding groupings, and c) defining new aggregate function. These features allow PDFOL to model person-centric databases, enabling formal and efficient reason about their accuracy.

This dissertation also explains how data analysts can use PDFOL statements to formalize and develop formal accuracy metrics specific to a person-centric database, especially if it is an integrated person-centric database, which in turn can then be used to assess the accuracy of a database. Data analysts apply these metrics to person-centric data to compute the quality-assessment measurements, Y^D. After that, they use statistical methods to compare these measurements with the real-world measurements, Y^R. Compare Y^D and Y^R with the hypothesis that they should be very similar, if the person-centric data is an accurate and complete representations of the real-world population.

Finally, I show that estimated accuracy using metrics based on PDFOL can be good predictors of database accuracy. Specifically, I evaluated the performance of selected accuracy metrics by applying them to a person-centric database, mutating the database in various ways to degrade its accuracy, and the re-apply the metrics to see if they reflect the expected degradation.

This research will help data analyst to develop an accuracy metrics specific to their person-centric data. In addition, PDFOL can provide a foundation for future methods for reasoning about other quality dimensions of PII. (83 Pages)

PUBLIC ABSTRACT

Estimating Accuracy of Personal Identifiable Information in Integrated Data Systems

Amani Shatnawi

Both government agencies and private companies rely on the collection of personal data on an ever-increasing scale. Out of necessity, person data include Personal Identifiable Information (PII), which is information that could potentially identify a specific individual. Many of these data would be integrated, so data analyst, policy makers or corporate officers can use it to make decisions or get a conclusion. Integrating data in a heterogeneous database environment create a need to estimate the accuracy of that data; without a valid assessment of accuracy there is a risk of coming with incorrect conclusions or making bad decision based on inaccurate data. Confidentially issues and the inaccessibility of the real individuals raises the question of how to measure the accuracy of person data, and specifically PII. So, the problem becomes one of estimating data accuracy using real-world facts, expert opinions, or aggregate knowledge about the represented population.

Estimating the quality of PII creates a corresponding need to model and formalize PII for both the real-world and electronic data, in a way that supports rigorous reasoning relative to real-world facts, rules from domain experts, and rules about expected data patterns. This research presents an extended first-order logic language (FOL), called PDFOL (Person Data First-order Logic), that can express these kinds of facts and rules, as well as relevant person attributes and inter-person relations. The salient features of PDFOL are: 1) namely temporal predicated based on time intervals, 2) aggregate functions, and 3) tuple-set comparison operators. I adapt and extend the traditional aggregate functions to allow any arbitrary number free variables in function statement, we add groupings feature to aggregate functions and we define new aggregate function. These features allow PDFOL to model person-centric databases, enabling formal and efficient reason about their accuracy and help to provide methods for reasoning about the accuracy of PII.

Also, I propose a method that describe how data analysts can use PDFOL statements to formalize and develop formal accuracy metrics specific to a person-centric database, especially if it is an integrated

person-centric database, which in turn can then be used to assess the accuracy of a database. Where data analysts apply these metrics to person-centric data to compute the quality-assessment measurements. After that, they statistically compare these measurements with the real-world measurements, with the hypothesis that they should be very similar, if the person-centric data is an accurate and complete representations of the real-world population.

I evaluated the performance of the developed accuracy metrics and their predicative capability and we prove that the developed accuracy metrics are applicable and easily can be used to estimate the accuracy of person-centric data. The evaluation presents how the proposed methodology can estimate the accuracy of the person-centric data and give an accuracy value is almost equal the real-accuracy with some deviations.

ACKNOWLEDGMENTS

First and foremost, I would like to express my gratitude and obligation to Allah for providing me the blessings to get my Ph.D. I could never have done this work without the mercy of Almighty, Omnipotent and Omniscient.

Then, it is a pleasure to thank the many people who made this dissertation possible. Initially, I would like to show gratitude and thanks my advisor, Dr. Stephen Clyde, who led my steps in this cumulative process. I could not imagine having a better supervisor for my Ph.D. He provided invaluable assistance, encouragement, good teaching, lots of good ideas, and support throughout the entire process. Definitely, writing this dissertation would have been extremely tough without his presence. I hope that he will support me morally and technically for the rest of my life.

Very special thanks go to my committee members, Dr. Curtis Dyreson, Dr. Xiaojun Qi, Dr. Nicholas Flann and Dr. Bedri Cetiner for giving me the extra strength and motivation to get things done.

I would like to thank my beloved parents and my parents in law for their supplications, love, constant prayers (doa's), and unconditional support throughout my life. My brothers and my sisters for their endless advises and support, love, encouragement, guidance, and unconditional affection that made all of this possible. Special thanks to all my friends for their encouragement and motivation that helped me reach here.

Last but not the least, I would especially like to thank my husband (Anas) for his unending support during the most challenging periods of my Ph.D. research. Thanks to my daughter and my son: Mayar and Omar, who have missed me a lot during my study.

Acknowledgment is due to Utah State University for its utilities, staff support, and professional activities from the time I came to the university. I would also thanks who supports me financially during my study, Dr. Stephen Clyde and Computer Science department at Utah State University.

Amani Shatnawi

CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER 1

INTRODUCTION

Both government agencies and private companies keep vast databases containing sensitive personal information about different individuals. Ideally, many of these databases would be integrated, so policy makers and corporate officers could make informed decisions. The need for the integration of data in a heterogeneous database environment creates a corresponding need for estimating the accuracy of that data. Without a valid assessment of accuracy there is a risk of data users coming to incorrect conclusions or making bad decision based on inaccurate data. In this research, we propose a theoretical method for developing data-accuracy metrics specific for any given person-centric integrated system. Also, we explain how a data analyst can use these metrics to estimate the overall accuracy of person-centric data.

A person-centric integrated system is a system that works primarily with person data from multiple data sources. Such systems exist in many different application domains, including the educational, medical, and judicial fields. Out of necessity, person data include Personal Identifiable Information (PII), which is information that can be used to identify, contact, or locate individuals. In databases, PII can be found in both attributes and inter-person relations. A *person attribute* is a characteristic of a real person and an *inter-person relationship* is an association among persons.

Data integration in the context of person-centric systems is a combination of technical and business processes that match, link, and merge the data from different sources to create concrete or virtual views of people, households, and organizations. Matching processes determine which records from various sources refer to the same individual, household, or organization. For individuals, they can fall into two broad categories: a) algorithms that establish identity using multiple PII elements and b) algorithms that use unique identifiers. For households, the matching process typically tries to identify individuals residing at same address. The process is similar for organizations, making use of organization identifying information or unique identifiers.

For data integration, matching is typically followed by linking, which is the construction of associations between records for the same entities across data sources. Mapping tables that represent cross-

database record references are a common mechanism for managing these associations. Some systems, like Utah's CHARM and the dohMPI, use a hub-n-spoke style of record mappings where all data sources records are mapped to central virtual identity [1].

After linking comes merging, which is the combination of data from different data sources to form the single view or "best version of truth" an entity. For example, imagine an integrated system with three data sources: A, B, and C. Each data source contains a record for some person, Joe. Each data source captures Joe's name, birth date, and address. Let us say that the names and birthdates in A are verified against legal documentations, like driver licenses, but that the data in A for a person is only updated about every 10 years on average. Also, imagine that B and C update their data for a person about every 6 months on average, but do not verify it against any legal documentation. A reasonable merging process would use Joe's legal name and birth date from A, but Joe's preferred name and addresses from B and C. Merging is typically driven by these kind of business rules, which consider the authenticity, verification, and timeliness of the source data. Unlike matching and linking, which usually take place on the front-end of data integration, merging can be done at any time after linking. For example, the merging for Joe could occur immediately after any changes in links among Joe's records or it could be deferred until there is a request for Joe's integrated data.

PII and the challenges associated with its management have become ubiquitous, especially as information technology and the Internet have made it easier to collect and disseminate protected personal data [1]. Confidentially issues aside, if these databases were integrated so more accurate and complete information about individuals were available, then policy makers, corporate officers, and professionals would be able to make informed decisions. Integrated person databases could help create real-time views of people and households, which in turn could help answer research questions, test hypotheses, and evaluate outcomes. However, the value of integrated data with respect to these activities depends on the accuracy of the data and the completeness of information about individuals.

The most direct way to measure data accuracy would be to compare the data with real-world people, individual by individual, attribute by attribute, relationship by relationship. Not only is this impractical in most cases because of its labor-intensive nature, but is often impossible because of the confidential nature of PII data and the inaccessibility of the real individuals. Nevertheless, many governmental agencies and companies are integrating person data, within the bounds of what is permissible by law and data sharing

agreements, and relying on that data for decisions. However, the benefits of integrated person data can only be realized if the data are accurate, relatively complete, timely, and pertinent [3]. This raises the question of how to measure the accuracy of person data, and specifically PII, when it is impractical or impossible to verify the person data directly with individuals.

Assuming that direct verification is not an option, the problem becomes one of estimating data accuracy using real-world facts, expert opinions, and aggregate knowledge about the represented population. Real-world facts are hard rules or constraints about person attributes or relations which should always be observed, e.g., a person's birthdate must be before or on his/her death date, if there is death date, or a child cannot be an ancestor of a parent. Violations of these constraints are clear indications of incorrect or incomplete data. Expert opinion are also rules, but these rules might have exceptions. For example, a person's birth date is typically more than 14 years after her/his biological mother's birth date. A violation of this is an indication of possible bad data. Finally, aggregate knowledge about a population can express norms or trends. Comparing the same kinds of aggregations for a database and real population can help data analyst's spot incorrect or incomplete data. For example, consider a database, like a birth registry, that is supposed to contain data all on children born in an area, e.g., a state or country. Also, assume that through census data and other studies, we know that 3.4% of the children born in that area are twins. If only 1.5% of children in the database are twins, then a data analyst could conclude that either the database is missing individuals or the twin information is inaccurate.

1.1 Research Objectives

Estimating the accuracy of PII in an integrated person-centric database depends on methods for modeling and formalizing PII for both the real-world and electronic data, in ways that support rigorous reasoning relative to real-world facts, expert opinions, and aggregate population knowledge. This research provides such a foundation by introducing a temporal first-order logic language (FOL), called *Person Data First-order Logic* (PDFOL), specifically designed to express person attributes and inter-person relations, as well as real-world facts, expert opinions, and aggregate knowledge. Chapter 2 provides some background on FOL's and Chapter 3 introduces the underlying concepts and syntax for PDFOL.

In general, FOL semantics are established by defining how their symbols map to mathematical structures, called models, consisting of objects and relationships [2] [3]. Formally, a mapping of a FOL's symbols to a single model is an "interpretation." However, if the structure of all possible models and the rules for mapping the FOL symbols to any given model are consistent, then we can simply think of a given model as an interpretation, without discussing the mapping. The set of all possible interpretations comprises the complete semantics for an FOL. Chapter 4 introduces a model structure for PDFOL and the rules for mapping the language's symbols to objects and relations defined by the structure. In other words, Chapter 4 establishes a theoretical foundation for PDFOLs semantics.

With its syntax and semantics formalized, PDFOL provides a mechanism for expressing data-accuracy metrics, computing measurements using these metrics on person-centric databases, and comparing those measurements with expected values from real-world populations. Figure 1.1 illustrates the overall approach. To begin, assume that a database *D*, represented by the drum icon on the top left, captures information about the persons in some population *R*, represented by the people icon on the top right. Both *D* and *R* change over time, but the two are not necessarily in sync. In fact, being able to assess how well *D* tracks *R* is the primary outcome of this research. The model icon labeled M^D represents a temporal PDFOL model for *D* and interpretation for PDFOL statements. Chapter 4 describes how M^D can be automatically generated from *D* and shows that PDFOL approach minimizes the number of tuples necessary to accurately and completely capture all the additions, changes, or deletions in the *D*.

The magnifying glass icon in the center of the figure represents facts, expert opinions, and aggregate knowledge. Data analysts formalize the facts and expert opinions by expressing them as closed PDFOL statements and the aggregate knowledge as open PDFOL statements. Chapter 5 describes how this is done. Next, as described in Chapter 6, data analysts use these PDFOL statements to develop metrics, illustrated by the icon in top center of the figure. Then, the data analysts apply these metrics to M^D to compute the quality-assessment measurements, Y^D. After that, they use statistical tests to compare Y^D with the real-world measurements, Y^R. The database accuracy is estimated by the correlation (or lack thereof) between Y^D and Y^R. See Chapter 7.

In practice, the real-world measurements, Y^R, come from surveys or census results. However, it is important to demonstrate that same process used to compute Y^D from *D* could be used to compute Y^R from *R*.

In Figure 1.1, M^R represents a PDFOL model that could be generated from R. See Chapter 4. As noted earlier, this process would be impractical for all but the simplest populations and may be prohibited by confidentiality regulations. Fortunately, it is not necessary to generate M^R. Instead, it is only necessary to show that M^R could be systematically generated and that it would be finite, for any finite time interval. Given this fact, it is then possible for data analysts to

1. Reverse-engineer Y^R into facts, expert opinions, and aggregate knowledge
2. Create PDFOL statements that accurately express these concepts
3. Develop metrics from the PDFOL statements
4. Use those metrics to compute quality-assessment measurements, Y^D
5. Compare Y^D and Y^R with the hypothesis that they should be very similar, if D is an accurate and complete representation of R

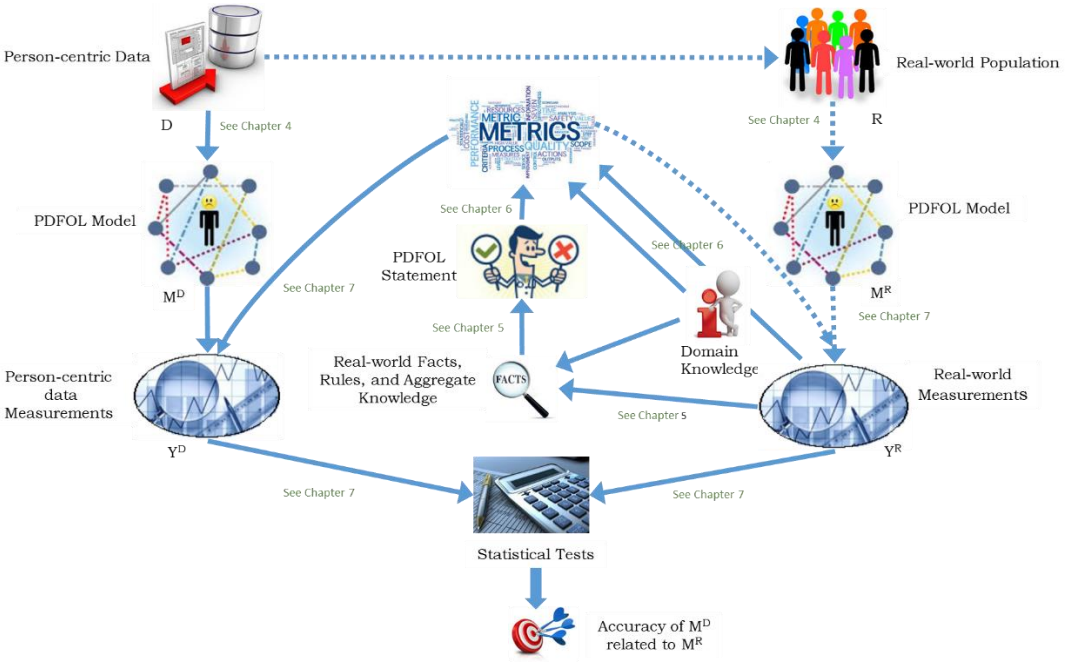


Figure 1-1.Research Overview

CHAPTER 2

FIRST-ORDER LOGIC OVERVIEW

First-Order Logics (FOLs) are formalisms for symbolic reasoning. In their most basic form, they consist of constants, variables, predicates, logical connectors, and quantifiers [4] [5]. More sophisticated FOLs also include functions and typed predicates [6]. Typically, the logical connectors include: conjunction, written as "and" or ⋀; disjunction, written as "or" or ⋁; negation, written as "not" or ¬; and implication, written as "implies" or ⇒. However, minimal FOLs can get by with only two connectors, such as ⋀ and ¬ [7]. The qualifiers usually include: universal, written as "for all" or ∀, and existential, written as "there exists" or ∃. For finite data sets, these quantifiers are equivalent to long conjunctive and disjunctive sentences [4] [5]. In general, FOLs model the world in terms of:

- Objects: things of interest, such as a person, a company, or a product, as well as properties of those things, such as a name, a tax identifier, or a description. Objects can also be compound objects, such as an address, which might be comprised of a street number, street name, city, state, and postal mailing code. For person data, the things of interest are people and their attributes or characteristic. Collectively, all the objects comprise a *universal domain*, also called the domain of discourse.
- Relations: sets of relationships or links among objects with domain-specific meanings, such as: *x is-bigger-than y*, *x is-outside-of y*, *x is-part-of* y, and *x owns y*. Every relation has an arity, which is the number of objects its links connect. It is also the number of columns, when representing a relation as a table. For person data, the interesting relations are those that link people with other people attributes, such as: *x is-a-sibling-of y* and *x is-the-first-name of y*.

Formally, a set of objects and relations forms a "model" and the mapping of symbols from the FOL to a model is called an "interpretation" [8]. The model is the universe about which the FOL is being used to reason. Because our aim is to reason about the quality of integrated person data, our models will be derived from temporal snapshots of integrated or federated person databases.

2.1 Benefit of FOLs

FOLs have several properties that make them a good choice for reasoning about person data in both the real-world and person-centric database. First, unlike natural languages, FOLs can support formal and automated reasoning. Given an interpretation, closed statements can be mechanically tested as valid or invalid against a model and open statements can be evaluated for valid bindings and thereby produce results like queries. Second, FOLs allow a variable to be bound to the objects in a model using existential and universal quantifiers, which provide a great deal of expressive efficiency compared to propositional logics. Third, FOLs with counting quantifiers [9] or aggregate functions are equivalent to relational algebras, meaning that queries expressed in a standard relational query language (SQL) can be translated into the FOL and any FOL statement can be translated to SQL [10].

2.2 FOL Syntax

In general, a first-order language is a set of non-logical and logical symbols. The non-logical symbols represent predicates (relations), functions and the constants on the domain of discourse. A predicate symbol with n-arity, where arity is the number of arguments and n >= 0. For example, "*IsASiblingOf* "is a 2-place predicate symbol. A function symbol with n-arity, and n >= 0. For example, "*TheFatherOfX*". Function symbols of 0-arity are called constant symbols. Section 2.3 explain the semantic meaning of relation and function and how they differ from each other.

Logical symbols include symbols, which always have the same meaning. For example, the logical symbol ∨ always represents "or"; it is never interpreted as "and". There are several logical symbols in the FOL, which vary by author but usually include quantifiers ∀, ∃, binary connectors ∧, ∨, →, not ¬, equality = and a set of variables. Below is an abbreviated Backus-Naur Form (BNF) syntax definition for a basic FOL:

<Statement> **::**= <Atomic Statement> |
<Unary Logical Connector> <Statement> |
<Statement> <Binary Logical Connector> <Statement> |
(<Statement>) |
<Quantifier> <Variable Binding> <Statement><Atomic Statement> **::**= <Predicate> (<Term> {“,“ <Term>})

<Variable Binding> ::= <Variable> | <Variable> "∈" <Domain>

<Domain> ::= *any named subset of the universal domain*

<Term> ::= <Function> (<Term> {"," <Term>}) | <Constant> | <Variable>

Constant ::= *any number or quoted string*

Variable ::= *any string beginning with a letter and containing only letters, digits, and underscores*

<Unary Logical Connector> ::= "¬"

<Binary Connector> ::= "∨" | "∧" | "⇒"

<Quantifier> ::= "∃" | "∀" <Predicate> ::= "Equal" | "GreaterThan" | *any other string that will name a relation*

<Function> ::= "Plus" | "Minus" | "Concat" | *any other string that will name a function*

Complete predicate statements are logically combined and manipulated according to the same rules as those used in Boolean algebra. Below is a summary of key concepts and rules.

1. A *term* is a constant symbol, variable symbol, or n-place function of n terms. For example, if f is an n-place function symbol (with $n \geq 0$) and $t_1, t_2, \ldots, t_n$ are terms, then $f(t_1, t_2, \ldots, t_n)$ is a term.
2. An *atomic statement* is an *n*-place predicate of *n* terms. For example, if *P* is an *n*-place predicate symbol (with $n \geq 0$) and $t_1, t_2, \ldots, t_n$ are terms, then $P(t_1, t_2, \ldots, t_n)$ is an atomic sentence.
3. A *complex statement* is a composition of one or more statements (atomic or complex) connected by the logical connectives. For example, if P and Q are statements, then $\neg P$, $P \vee Q$, $P \wedge Q$, and $P \rightarrow Q$ are all complex statements.
4. A quantified statement binds a free variable in a statement using a *universal quantifier*, ∀, or a *universal quantifier*, ∃. For example, the statement ∃y P(y) means "There exists some *y* such that *P* ("joe", *y*) is true." The statement $\forall x \exists y\ P(x, y)$ means "For all x, there exists some *y* such that *P* ("joe", *y*) is true."
5. A close statement is a statement containing no "free" variables. In other words, all variables are "bound" by universal or existential quantifiers. For example, $\forall x \exists y\ P(x, y)$ is a closed statement, but $\forall x\ P(x, y)$ is not.

2.3 FOL Semantics

Interpretations establish the semantics of an FOL by defining the possible meanings of the languages symbols. In general, constants in FOL statements map to objects in a model. The logical connectors and quantifiers have fixed meanings, which are the same for all interpretations and are typically consistent across FOLs. The meaning of a variable is determined by the quantifier that binds it, if there is one. If there is no binding quantifier, then the variable is "open" to any interpretation.

Every predicate has some number of places and maps to relation in a model with that exact arity. For example, if *IsASiblingOf* were a 2-place predicate symbol, an interpretation would map it to a binary relation that represent *x is-a-sibling-of y*. A FOL statement using this predicate would be written as *IsASiblingOf (x, y).*

Like predicates, functions have arity and map to relations. However, where an *n*-place predicate maps to an *n*-ary relation, an *n*-place function maps to a relation that has an arity strictly greater than *n*. The extra columns in the relation correspond to the output of the function. For example, consider a 2-place subtraction function, called *Minus*. It might map to a 2-place relation, *r*, where the first place of Minus corresponds to the first place of *r*, the second place of Minus maps to the 2nd place of *r*, and the output of *Minus* maps to the 3rd place.

To lay a foundation for the definition aggregation functions in PDFOL (see Chapter 3), we loosen the typical definition of a function to allow an element of its domain to map to multiple elements of the range, so the function can yield a set of value or tuples instead of a single value. For interpretations, this means that functions only differ from predicates in how their mappings are formed. A function can map to any relation in a model, provided the relation has an arity that is greater than the number of places of the function.

2.4 Open and Closed Statements

In general, there are two kinds of statements, closed and open, that differ in how they use variables. In closed statements, all variables are bound by quantifiers. Within a system like a person database, close statements define facts or data. Open statements include one or more unbounded or free variables and therefore are "open" to interpretation, so they can be used to reason about questions or make queries. When

evaluating an open statement, the result is a set of tuples containing all the valid ("true") bindings for the free variables.

Our loose definition for functions allows open statements to be re-cast as functions, where the output of the function is a query result. For example, consider a system that includes a predicate, called *HasProfession* that represents an *x*-has-*a-profession-of-y* relation. Also, assume the system includes the following facts:

HasProfession("Joe", "Teacher")

HasProfession("Mary, "Teacher")

HasProfession("Sue", "Administrator")

The evaluation of the open statement *HasProfession(x, "Teacher")*, would yield *{"Joe", "Mary"},* which are the possible valid bindings for *x*. We can define a 1-place function, *WhoHasProfession*, such that *WhoHasProfession*(y) = *HasProfession(x, y)* and the evaluation of *WhoHasProfession ("Teacher")* would yield the same set as the original open statement.

As explained above, some functions, such as *WhoHasProfession*, return sets that may contain more than one object. We call these functions, *set functions*. Other functions, called *singleton functions*, always return a set with exactly one object. For singleton functions, we will allow the object in the set to be treated as the function output.

2.5 Substitution

When working with statements in a FOL, it is often necessary to manipulate them by substituting variables in a statement with other terms, like constants that represent objects in the model. For example, let *S* be an open statement defined as:

S = HasProfession(x, y)

To evaluate it for "Joe" and "Teacher", we need transform this statement into a closed statement, by substituting "Joe" for *x* and "Teacher" for *y*. We will use the following notation to express the substitution of one or more variables [11]:

$S_{\{v_0 \mapsto h_0, \ldots, v_k \mapsto h_k\}}$ or $S_{\{v \mapsto h\}}$

where $v_0, \ldots, v_k$ or v, for a shorthand, are variables and $h_0, \ldots, h_k$ or h are the terms that the will replace the variables. For example,

$$S_{\{x\mapsto\text{"Joe"},y\mapsto\text{"}Teacher\text{"}\}}$$

yields the statement *HasProfession("Joe", "Teacher")*.

2.6 Logics with Counting

FOLs have the ability to express facts about how many objects have a certain property, where these quantifiers can be expressed in terms of cardinality and can be defined in terms of absolute count [12] [13]. For example, the ideas "no professor supervises more than 3 graduate students" and "every graduate student is supervised by at most 1 professor" maybe formalized as follows:

$$\neg\exists x(prof(x) \wedge$$

$$\exists y_1 \ldots y_4\Big(\bigwedge_{1\le i\le 4}(grad(y_i) \wedge sup(x,y_i)) \wedge \bigwedge_{1\le i,j\le 4\ and\ i\neq j}(y_i \neq y_j)\Big)$$

$$\forall x\, y_1 y_2(grad(x) \wedge prof(y_1) \wedge sup(y_1,x) \wedge prof(y_2) \wedge sup(y_2,x) \rightarrow y_1 = y_2)$$

A more succinct and readable formalization is possible by adding limits to the familiar quantifiers ∀ and ∃, making them counting quantifiers. These counting quantifiers are written as $\exists_{\le U}$, $\exists_{\ge L}, \exists_{=C}$ *or* $\exists_{\ge L:\le U}$ where U, L, and C are integers [12], and it can be read as:

- $\exists_{\le U} v\, \mathcal{F}(v)$ → "there exist at most U of v's such that $\mathcal{F}(v)$"
- $\exists_{\ge L} v\, \mathcal{F}(v)$ → "there exist at least L of v's such that $\mathcal{F}(v)$"
- $\exists_{=C} v\, \mathcal{F}(v)$ → "there exist exactly C of v's such that $\mathcal{F}(v)$"
- $\exists_{\ge L\le U}\, v\, \mathcal{F}(v)$ →" there exist between L and U of v's such that $\mathcal{F}(v)$"

Using counting quantifiers, the "no professor supervises more than 3 graduate students" and "every graduate student is supervised by at most 1 professor" concepts can be captured more concisely as follows:

$$\forall x(\text{prof}(x) \rightarrow \exists_{\le 3} y(\text{grad}(y) \wedge \text{sup}(x,y)))$$

$$\forall x\big(\text{grad}(x) \rightarrow \exists_{\le 1} y(\text{prof}(y) \wedge \text{sup}(y,x))\big)$$

When a model (used in interpreting FOL statements) is finite, then statements with counting quantifiers can be mapped to equivalent statements without a counting quantifiers, but with many more clauses and

variables. So, in the context of finite domains, counting quantifiers do not change expressive power as FOL; they are simply a notational convenience.

2.7 First-Order Temporal Logic

First-order Temporal Logic (FOTL) is used to reason about statements whose truth depends on time [14] [15]. FOTL can be used as a natural temporal query language for point-stamped temporal data. A query (a temporal logic formula) is evaluated with respect to an evaluation point (time instant). Each such point determines a specific data snapshot that can be viewed as an instant.

Each relation R of arity k in the database schema can be extended with a timestamp representation contains the linear order on the states [14] [15]. The extended relation $\bar{R}$ of arity $k+1$ holds the data element in the first k columns and the last column holds timestamp. Although these approaches allow for temporal reasoning, the models grow non-linearly with the passage of time. When the models stem from real databases, this is an awkward characteristic because it can restrict how and when temporal snapshots are recorded.

A different approach uses time intervals consisting of inclusive starting times and exclusive ending times, written as *(t1, t2]* where *t1* is the starting time and *t2* is the ending time [15]. With such approach, models grow linearly with respect to changes, and do not grow with just the passage of the time. Therefore, this approach is more practical when working with models derived from real data or models that have fine-grain or even infinitely dense time.

An alternative way of providing a temporal query language is to extend first-order logic with temporal operators, such as sometime in the future or until, by going around the set of temporal connectives [14] [15]. Temporal operators will provide implicit references to time instants rather than explicit time-variables. S. Abiteboul et al., [14] propose a work that presents an extension of the FOTL. They use temporal operators based on the regular events that are leading to the Extended Temporal Logic (ETL) [16]. Temporal logic in the form of ETL fragment having as only temporal operators since, until, next, and previous; where the syntax of these fragment of ETL can be obtained by using the formation rules for the standard first-order logic over the same database schema with one additional formation rule. When searching for a regular event in the future, then use the (L^{+}) connective, which reaches precisely the last state of the temporal database.

J. Chomicki and D. Toman [15] propose a way to define the temporal query by using the FOTL, where the answers to these queries are (valid-time) point-stamped temporal relations. During their work, they defined the commonly used temporal connectives, such as sometime in the future, always in the future, sometime in the past, and always in the past in terms of since and until connectives [7].

CHAPTER 3

PERSON DATA FOL

PDFOL is an FOL with extensions for temporal predicates, aggregate functions, and comparison predicates. We aim to keep PDFOL simple, yet expressive for person data and population-wide statistics. To this end, PDFOL makes few changes and extensions in the typical FOL syntax that allow for an in-line comparison predicates and the addition of aggregate functions. Below is a definition of PDFOLs grammar, using an abbreviated Backus-Naur Form [17]. The semantics for PDFOL are defined by mappings to arbitrarily large, but finite models. See Chapter 4.

<Statement> **::**= <Atomic Statement> |
<Unary Logical Connector> <Statement> |
<Statement> <Binary Logical Connector> <Statement> |
(<Statement>) |
<Quantifier> <Variable Binding> <Statement> |
<Comparison Statement>

<Atomic Statement> **::**= <Predicate> (<Term> {"," <Term>})

<Variable Binding> **::**= <Variable> | <Variable> “∈” <Domain>

<Domain> **::**= any named subset of the universal domain

<Term> **::**= < Function> (<Term> {"," <Term>}) |
<Aggregate Function(<Term> {"," <Term>}) |
<Constant> | <Variable>

Constant **::**= *any number or quoted string*

Variable **::**= *any string beginning with a letter and containing only letters, digits, and underscores*

<Unary Logical Connector> ::= "¬"

<Binary Connector> **::**= "∨" | "∧" | "⇒"

<Quantifier> **::**= "∃" | "∀"

<Predicate> **::**= "Equal" | "GreaterThan" | *any other string that will name a relation*

<Function> **::**= "Plus" | "Minus" | "b" | *any other string that will name a function*

<Aggregate Function> ::= "Sum" | "Min" | "Max" | "Count" | "Projection"

<Comparison Statement> ::= <Term> "==" <Term>

3.1 Temporal Predicates

In general, a Temporal FOL (TFOL) describe systems where objects and relationships exist at specific times and can allow users to reason about facts over time [15]. For example, E. Mendelson [5] and R. A. Freire [6] developed TFOLs by adding single time points to relations and by introducing temporal predicates, such as "before" and "after". Although these approaches allow for temporal reasoning, the models grow non-linearly with the passage of time. When the models stem from real databases, this is an awkward characteristic because it can restrict how and when temporal snapshots are recorded.

OSM-Logic [18] takes a different approach by using time intervals consisting of inclusive starting times and exclusive ending times, written as $(t_1, t_2]$, where t_1 is the starting time and t_2 is the ending time. With this approach, models grow linearly with respect to changes, and do not grow with just the passage of the time. Therefore, this approach is more practical when working with models derived from real data or models that have fine-grain or even infinitely dense time. PDFOL adopts OSM-Logics approach, but restricts time to a finite set of ordered points. As long as the time intervals can be fine-grain (smaller than the smallest frequency of change), this approach is sufficient for person data.

Valid interpretations of PDFOL map every temporal predicate to a relation *r* that has two extra places that represent the time intervals. Consider a model that includes the time points {1, 2, 3, 4, 5} and the temporal relation for persons and their income shown in Table I. The first three places represent a person, income source, and income. The last two places are the starting and ending times.

As with OSM-Logic, temporal relations in PDFOL models must all adhere to following time constraint: if a temporal relation, *r*, contains a tuple linking objects $x_1, \ldots, x_m$ and a time interval of $(t_i, t_k]$ and there exists some t_j, where $t_i < t_j < t_k$, then *r* must contain two other tuples with same objects $x_1, \ldots, x_m$ and the time intervals $(t_i, t_j]$ and $(t_j, t_k]$. The additional tuples are referred to as secondary tuples because they are implied by first tuple. Table 3.1 only shows the primary tuples, which cannot be implied by other tuples.

All predicates in PDFOL are temporal predicates and therefore mapped to temporal relations in an interpretation. Also, object places in a predicate are mapped to object columns in the underlying relation and time places are mapped to the corresponding time columns of that relation. In this way, PDFOL is like a simple two-sorted FOL [19]. For a temporal predicate clause to be true, the relation corresponding to the

predicate must contain a tuple that relates the objects to each over the entire time interval specified in the clause.

In PDFOL, a function is defined with an open statement or set-building expressions, where the places of the function are bound to free variables of the statement and the statement includes at least one other free variable. The results of evaluating a statement using a function is the set of valid bindings for the remaining free variables in the functions definition statement. Because such definition statements capture the meaning of the functions, PDFOL interpretations do not explicitly map functions to relations in the model.

3.2 Aggregate Functions

In general, an aggregate function *AF* applies to a subject statement *S* with free variables and is written as $AF_{X|G|F}<S>$. The free variables of S are the subject variables (*X*), grouping variables (*G*), and other free variables (*F*). Including aggregate functions in FOLs is not a new idea. For example, Natsev, et al., describe *Sum*, *Max*, *Min*, and *Count* functions [10] [20]. However, we adapt and extends these traditional aggregate functions in three ways: a) allowing any arbitrary number free variables in S, b) adding groupings, and c) defining a new aggregate function, namely *Projection*.

- For *Min*, *Max*, and *Sum*, *X* must contain a single free variable in *S*. For *Count*, *X* is a set of one or more free variables and, for *Projection*, *X* must be empty. When *X* is non-empty, its variables are the subject of the function. In other words, it is what the function is counting, summing, etc.
- The grouping variables, *G*, may be an empty or non-empty set for any of the functions. If it is not empty, its variables represent grouping criteria for the results, like a GROUP BY clause in SQL. More specifically, if *G* is non-empty, then results for *AF* contain one tuple for each distinct set of objects bound to the *G* variables.
- *F* must contain all other free variables of S that are not *X*, or in *G*. When F is not empty, it is equivalent to binding those variables with existential qualifiers in S.

Some aggregate functions are singleton functions; others are set functions. Specifically, any *Projection* function or aggregation function with a non-empty *G* is a set function. All others are singleton functions.

All the tuples in the result of a set function will have the same *n* places. For *Projection*, $n = |G|$. For all others, $n = |G| + |X|$. Therefore, a set functions result can also be treated as relation and be represented by an *n*-place predicate. Consequently, a set function can be used in a *Predicate Clause* in a statement.

Tables 1-3 show three relations for a sample model that we will use to illustrate PFDOL and its aggregate functions. For this model, assume that 5 time points are {1, 2, 3, 4, 5}.

Also, for our examples, we will use the predicates: *Income*, *MotherOf*, and *MotherRace*. The *Income* predicate has place roles of (*Person*, *Source*, *Income*, *StartTime*, *EndTime*). Below is an example of an open statement using this predicate that would query the whole *person-earns-income* relation.

$$Income(p, \mathrm{c}, i, t_1, t_2)$$

Similarly, the *MotherOf* predicate has place roles of (*Person*, *Child*, *MotherRace*, *StartTime*, *EndTime*) and the *MotherRace* predicate has (*Person*, *MotherRace*, *StartTime*, *EndTime*).

Table 1. Person Earned Income Relation

Person	Source	Income	Start Time	End Time
Jolley	A	1500	1	2
Jolley	B	1600	2	5
John	A	1000	2	3
John	C	1010	2	4
Mathew	D	2000	2	4
Ben	A	1700	2	3
Aaron	A	1400	2	4
Amy	A	1200	1	3
Amy	A	500	1	2

Table 2. Mother of Child Relation

Mother Name	Child Name	Start Time	End Time
Jolley	Ben	1	3
Jolley	Sam	1	5
Jolley	Ben	4	5
Sally	Suzy	2	5
Sally	Mathew	2	5
Sally	Randy	2	5
Lucy	Tala	1	4
Lucy	Luis	1	4
Rachel	Dana	2	3

Table 3. Mother Has Race Relation

Mother Name	Mother Race	Start Time	End Time
Jolley	Hispanic	1	5
Jolley	White	2	3
Sally	White	2	5
Lucy	White	1	4
Rachel	Hispanic	2	3

Tables 1-3, along with all their implied rows, will be the model for the examples and the interpretation will map the *Income* predicate to the *person-earned-income* relation, the *MotherOf* predicate to the *mother-of* relation, and the *MotherRace* predicate to the *mother-has-race* relation.

3.2.1 Sum function (sum)

The purpose of the *sum* function is to add up certain numeric objects from the model so the totals can be used in reasoning about the quality of the data in a model. For all instances of the *sum* function, *X* must contain a single free variable, *x*, that is in *S* and represents what will be summed by the function. Depending on whether *G* is empty, a *Sum* function can be either a singleton or a set function. As mentioned,

the *F* variables must be all other free variables in *S* that are not in *X* or *G*. Throughout the rest of this paper, we will use β to represent a distinct set of values for the *F* variables.

When used as a singleton-function, *sum* returns the sum of all possible values for *X* that will make *S* a true statement in the interpretation. More formally,

$$sum_{x||F} < S > = \sum \{h | \exists \beta (S_{\{x \mapsto h, F \mapsto \beta\}})\}$$

When used as a set function, *sum* returns a set of tuples, where each tuple has |*G*| + 1 places and the meaning of these places corresponding to the meaning of the *G* variables plus the sum of *X* over distinct values for those variables. A tuple in the result is (α, $\sum x$), where α is a set of distinct values for *G* such that *S* is true for some *x* and $\sum x$ is the sum of the all *x* values for which S is true given α. More formally,

$$sum_{x|G|F} < S > = \left\{ \begin{matrix} (\alpha, \sum \{h | \exists \beta (S_{\{x \mapsto h, G \mapsto \alpha, F \mapsto \beta\}})\}) \mid \\ \exists j, F(S_{\{x \mapsto j, G \mapsto \alpha, F \mapsto \beta\}}) \end{matrix} \right\}$$

To analysis the effects of *X*, *G*, and *F* for any aggregation function, we categorize their uses into 12 representative cases based on what is in *X* and *G*. See Table 4. When *G* is non-empty, it interesting to whether those variables represent objects, start time, and/or end time.

Although all these combinations are possible, they are not all interesting when analyzing the quality of person data. So, for space considerations, we only illustrate the cases marked with †, which are relevant to the purpose of this paper. We give an example of each these cases for the Sum function below.

Case1: *X* is not empty, *G* is empty. Using the *Income* predicate, the following statement sums all income at time 1, regardless of the person or source. Note that because PDFOL uses times that are inclusive at the start and exclusive at the end, the interval [1, 2) represents exactly time 1.

$$sum_{i||p,c} < Income(p, \mathrm{c}, i, 1, 2) >$$

The result of evaluating this function using the sample interpretation is 3200, because there are only three tuples in *person-earned-income* that have a [1, 2) time interval and they contain incomes of 1500, 1200, and 500.

Case 2: *X* is not empty and *G* includes objects. The following statement queries sums of income per person at time 1, regardless of income source.

$$sum_{i|p|c} < Income(p, c, i, 1,2) >$$

Table 5 shows the results of evaluating this statement. The only persons who earned income at time 1 are Jolley and Amy.

Case 3: *X* is not empty and *G* include objects, start time and end time. Below is an example that queries a person income for each time interval, where the money was earned from source *A*:

$$sum_{i|p,t_1,t_2|} < Income(p, "A", i, \mathrm{t}_1, \mathrm{t}_2) >$$

Table 6 shows the results of evaluating this statement. The persons with income from source *A* are Jolley, Amy, John, Ben, and Aaron and the time intervals during which they earned income from source A are [1, 2), [1, 3), [2, 3), [2, 4), and [3, 4). Note that Table 6 does not show any rows with [1, 2) and [3.4) because they are implied from rows with [1, 3) and [2, 4). Note that the result includes two rows for Amy, because she earned 1200 and 500 during the time [1, 2) and just 1200 during the time [2, 3).

Table 4. Free Variables in AF Statement

Free variables in AF statement				
X	*G*			Valid Syntax
	Object(s)	Start time	End time	
¬Empty †	Empty			Yes
¬Empty †	Y			Yes
¬Empty	Y	Y		Yes
¬Empty	Y		Y	Yes
¬Empty †	Y	Y	Y	Yes
¬Empty †		Y	Y	Yes
Empty	Empty			No
Empty	Y			No
Empty	Y	Y		No
Empty	Y		Y	No
Empty	Y	Y	Y	No
Empty		Y	Y	No

Table 5. Person Total Income at Time 1

Person	Income
Jolley	1500
Amy	1700

Table 6. Person Income Grouping with Person, Start and End Time

Person	Start Time	End Time	Income
Jolley	1	2	1500
John	2	3	1000
Ben	2	3	1700
Aaron	2	4	1400
Amy	1	2	1700
Amy	2	3	1200

Case 4: *X* is not empty and *G* includes start time and end time. Below is an example of statement that queries the total income of each time interval where the money was earned from source *A* by at least one person:

$$sum_{i|t_1,t_2|p} < Income(p, "A", i, \mathrm{t}_1, \mathrm{t}_2) >$$

See Table 7 for the results. The intervals during which incomed was earned from *A* are [1, 2), [1, 3), [1, 4), [2, 3), [2, 4), and [3, 4). The total income for a time interval [a, b) is the sum of all incomes that come from *A* during [a, b), keeping in mind that *person-earned-income* includes secondary tuples not shown in Table I. Also, a person, *p*, only earns an income *i* for a time interval [*a*, *b*) if *p* and *i* are associated over all [*a*, *b*). For example, during the interval [2, 3), John earned 1000 from *A*, Ben earned 1700, and Aaron earned 1400. However, none of them three earned income from *A* for the whole [1, 3) interval. For this reason, the result shown in Table 7 may seem counter intuitive, but the query is not asking for total money that *A* paid out at any point in a time interval. Instead, it is asking, for each time interval, what is sum of money paid to individuals throughout that entire interval. The former idea can be expressed with a different PDFOL query

that first gathers the amount of income for each time point, [a, a+1), and then sums those amounts together. Queries like case 4, on the other hand, are useful for looking at patterns over various time intervals.

3.2.2 Max function (max)

Like the *sum* function, the *max* function requires that X contains a single variable, *x*, but instead of adding up all the possible values for *x*, it finds the maximum value for *x*. If *G* is empty, the *max* function is defined as follows:

$$max_{x||F} < S >= h \mid \exists \beta \left(S_{\{x \mapsto h, F \mapsto \boldsymbol{\beta}\}} \wedge \neg \exists \mathrm{j} \left(j > h \ \wedge S_{\{x \mapsto j, F \mapsto \boldsymbol{\beta}\}} \right) \right)$$

The following sample statement returns the maximum income earn by a person for any time interval, and based on Table I that is 2000.

$$max_{i||p,c,t_1,t_2} < Income(p, c, i, t_1, t_2) >$$

When *G* is non-empty, *Max* returns a set of tuples, where each tuple has |*G*|+1 places, like the *Sum* function. The last place is the maximum *x* for a distinct set of values α for *G*.

$$max_{x|G|F} < S >= \{ (\alpha, h) \mid h = Max_{x||F}(S_{\{G \mapsto \alpha\}}) \}$$

For example, the following statement queries the maximum income of each person, regardless of income source and time.

$$max_{i|p|c,t_1,t_2} < Income(p, c, i, t_1, t_2) >$$

Table 8 shows the results of evaluating this statement, which is a 2-place table the roles {*Person*, *Income*}.

Table 7. Income Grouping by Start Time and End Time

Start Time	End Time	Income
1	2	3200
1	3	1200
2	3	5300
2	4	1400
3	4	1400

3.2.3 Min function (min)

Aggregate *min* function is like the *max* function, except that is results minimum values instead of maximum values. For the singleton-function form, its definition is as follows:

$$min_{x||F} < S > = h \mid \exists \beta \left(S_{\{x \mapsto h, F \mapsto \boldsymbol{\beta}\}} \wedge \neg \exists \mathrm{j} \left(j < h \ \wedge S_{\{x \mapsto j, F \mapsto \boldsymbol{\beta}\}} \right) \right)$$

For its set-function form, its definition is as follows:

$$max_{x|G|F} < S > = \{ (\alpha, h) \mid \ h = max_{x||F} \left(S_{\{G \mapsto \alpha\}} \right) \}$$

3.2.4 Count function (count)

As the name suggests, the *count* function counts the number of unique binding of the *X* variables that make *S* true. To express the definition for count using a set builder notation, we need to introduce a counting quantifier, $\exists^n h$, which means that there are exact *n* distance *h* values for which the subject sentence is true. For finite models, counting quantifiers can translated to longer conjunctions of statements with existential quantifiers [18].

For its singleton-function form, the definition for the count function is as follows:

$$count_{X||F} < S > = n \mid \exists^n \gamma, \beta \ \ (S_{\{x \mapsto \gamma, F \mapsto \beta\}})$$

Using the *MotherOf* predicate as an example, the following statement returns the number of children who have a mother with name "*Jolley*", regardless of the time interval, which is 3.

$$count_{c||t_1,t_2} < MotherOf(\text{Jolley}, c, \mathrm{t}_1, \mathrm{t}_2) >$$

For its set-function form, count is defined as follows:

$$count_{X|G|F} < S > = \{ (\alpha, h) \mid \ h = count \left(S_{\{G \mapsto \alpha\}} \right) \}$$

The following statement, for example, computes the number of kids for each mother, independent of the time interval.

$$count_{c|m|t_1,t_2} < motherOf(m, c, t_1, t_2) >$$

Table 9 shows the results of evaluation this statement. The table has 2-columns with place roles of {*Mother Name*, *Number of kids*}.

Table 8. Person Income

Person	Income
Jolley	1600
John	1010
Mathew	2000
Ben	1700
Aaron	1400
Amy	1200

Table 9. Mother and Number of Kids

Mother Name	Number of kids
Jolley	3
Sally	2
Lucy	2
Rachel	1

3.2.5 Projection function (proj)

The *proj* function is like a projection function in relational algebra [21] in that it returns a subset of the columns of a table. As we mentioned above, for all instances of the *proj* function, *X* must be empty and *G* must be non-empty. So, it only has a set-function form and its definition is as follows:

$$proj_{|G|F} < S > = \{\alpha | \exists \beta \ (S_{\{G \mapsto \alpha, F \mapsto \boldsymbol{\beta}\}})\}$$

For example, the following statement returns the set of mothers and their races regardless of time interval, and results in a 2-column table with place roles of {*Mother Name*, *Mother Race*}. Table 10 shows the results of this statement.

$$proj_{|m,r|t1,t2} < MotherRace(\mathrm{m}, r, t_1, t_2) >$$

3.3 Comparison Predicates

Intuitively, a comparison predicate is one that represents equality or inequality relation between individual objects, tuples, or sets of tuples. Formally, they are interpreted as non-temporal binary relations with only object places. PDFOL supports the following the ≤, ≥, <, >, =, and ≠ comparison predicates for ordered objects (like numbers and strings) and just = and ≠ for unordered objects, tuples and sets of tuples. We assume that they all map to relationships that represent their common meaning for numbers, strings, tuples, sets, etc. For readability, we allow comparison predicates to written in-line, e.g. $x \neq y$ instead of ≠(x, y). Below is a statement that is both syntactically acceptable in PDFOL and well-defined in terms of interpretation that says the number of mother race over the [1, 3) time interval should be the same over time interval [3, 5).

$$count_{r|m|t_1,t_2} < MotherRace(m, r, 1, 3) > = \ count_{r|m|t_1,t_2} < MotherRace(m, r, 3, 5) >$$

3.4 Summary

This chapter introduced an extended FOL, called PDFOL to express relevant person attributes, inter-person relations and the different kinds of constraints and rules (see Chapters 4 and 5). It differs from other FOLs in its approach to temporality and the addition of aggregate functions and in-line comparison predicates. Furthermore, its aggregate functions allow any arbitrary number of free variables in the function statement and groupings. These features allow PDFOL to model person-centric databases effectively, while enabling formal and efficient reason about their quality.

Table 10. Mother and Her Race

Mother Name	Mother Race
Jolley	Hispanic
Jolley	White
Sally	White
Lucy	White
Rachel	Hispanic

CHAPTER 4

MODELING PERSONS DATA

4.1 Person Data Modeling

A population is a set of *persons* with *attributes* and *relationships*. If all a person's attributes were known, they would distinguish her/him from all other persons. Each type of attribute has a *domain* of possible values. For example, a gender attribute may have the domain containing the values: male, female, and undetermined. Formally, we defined a population model as:

$M = (P, T, D, A, R, PA, PR)$, where

$P = \{p\}$ is finite set of people that comprise a population.

$T = \{t\}$ is a non-empty, finite, and ordered set of discrete time points. The last time point in T is always ∞, meaning an arbitrary time point some in the future.

$D = \{d\}$ is a universal domain that contains all attribute values, such as first name, last name, and birth date. The universal domain is arbitrarily large, but finite. Also, the elements of D are also finite in length or size. The universal domains can be divided into any number of domains for attribute definitions (see below). These attribute domains may be overlapping.

$A = \{a\}$ is a non-empty, finite set of attribute types. Each attribute type has:

Name, a.name

Arity, a.arity: the arity defines the number of places in the attribute. For a person attribute type, this is a person and a number attributes that involved in that type of person attribute plus 2 extra places represent the starting and ending time when relation existing in the population. For example, a composite attribute "*IncomeandSource"* would have an arity of 5.

*Attribute Domains, a.*domains, $\{a.d_i\}$, where $1 \leq i \leq a.arity\text{-}3$ and $a.d_i \subseteq D$.

Roles, a.roles which are a countable ordered set of phrases of size *a.arity* that specify the semantics of each place:

$$a.roles = (Person, r_i, \ldots, Start\ Time, End\ Time \mid 1 \leq i \leq a.arity\text{-}3 \wedge r_i \text{ is a noun phrase})$$

For example, the *"IncomeAndSource"* attribute type would have the roles: (*Person*, *Income*, *Source StartTime*, *EndTime*)

Sorts, *a.sorts:* is a countable ordered set of size $a.arity$ that contains the types of the roles of either *Person*, *Domain* or *Time*.

The type of the first place is *Person* and the last two places of any attribute type have a sort of *Time* while all the other places are $a.d_i$.

$$a.sorts=(P, s_i, \ldots, Time, Time \mid 1 \leq i \leq a.arity\text{-}3 \wedge s_i \subseteq a.d_i)$$

R={r} is a non-empty, finite set of person relation types. Each relation type has:

Name, r.name

Arity, *r.arity:* the arity defines the number of places in the relation. For a person relation type, this is number people are involved in that type of relation plus 2. The extra 2 places represent the starting and ending time when relation existing in the population. For example, a *"MotherOf"* relation type would have an arity of 4.

Roles, *r.roles*: a countable ordered set of phrase, of size *r.arity*, that specify the semantics of each place:

$$r.roles = \{r_1, r_2, r_3, \ldots, r_n : n \geq 4\}$$

For example, the *"MotherOf"* relation type would have the roles: (*Mother*, *Child*, *StartTime*, and *EndTime).*

Sorts, *r.sorts:* is a countable ordered set of size $r.arity$ that contains the types of the roles of either *Person* or *Time*.

The type of the last two places of any relation type have a sort of *Time* and all the other places is *Person*.

PA= {pa} is a set of n-place temporal relations, one for each attribute type *a* in *A*. each relationship *pa* has:

pa.type = a

pa.data = a relation of the sort *a.sorts*

PR= {pr} is a set of *n*-place temporal relationships, one for each relation type *r* in *R*. Each relationship *pr* has:

pr.type=r

pr.data = a relation of arity r.arity and with tuples follow r.sorts

In this definition, *P* and *D* are the objects of the model, *T* is the time structure, *PA* and *PR* are the relations, and *A* and *R* are meta-data about those relations. A population model can represent either a real-world population or a database that is supposed to contain information about a real-world population. To distinguish between the two kinds of populations, we will use the notation M^R to represent a model of a real-world population and M^D to represent a person-centric data. As you can see in Figure 4-1.

4.2 Real-world Population Model (M^R)

It is possible to represent a population model that captures the evolution of a real-world population by tracking the relevant changes to the people. For such a model, *T* would include a special time ∞ that represent a point in the future. Persons would correspond to objects in *P* and all other objects, such as person name, height, weight and birthdate, would be in *D*. Relationships between persons and their attributes would be captured in *PA* and relationships among persons would be captured in *PR*.

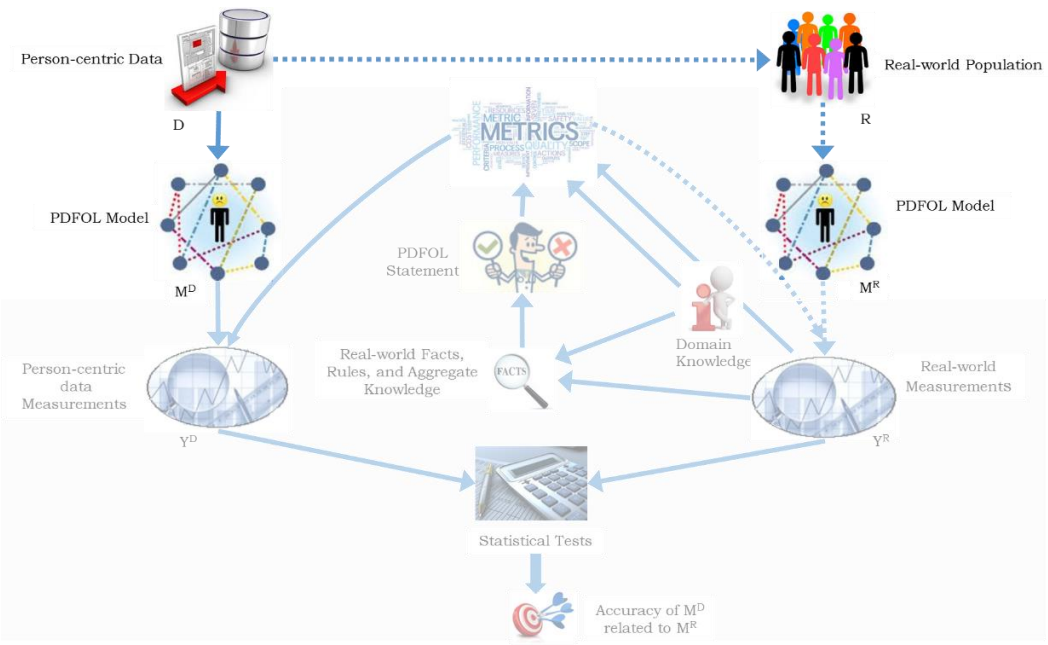


Figure 4-1. Modeling Personal Data

According to changeability of person's attributes and relationships among them. Each *pa* in *PA* and *pr* in *PR* has a time interval that gives the live time of the giving relationship. Whenever, a new person is added to the population at time t_1, then person would be added to *P* and his/her attributes would be added to *D*. The relationship between the person and its attributes would be presented by *pa* with time interval of $[t_1, \infty)$, and the relationships of him/her with other existing would be presented by *pr* with the same time interval of $[t_1, \infty)$. If a person attribute or relationship is changed at time t_2, then the time interval of corresponding relation *pa* or *pr* would be changed from $[t_1, \infty)$ to $[t_1, t_2)$, and the new attribute or relationship with the new values would be presented as *pa* or *pr* with an interval of $[t_2, \infty)$.

The following example shows how a model M^R could represent a real-world population. John is a white American man. He was born in Logan in May 5th, 1976 and lived in it until end of 2016. In the beginning of 1997, he moved to SLC to finish his study. He lived in Salt Lake City (SLC) until he got a Ph.D. in political science from the University of Utah in May 2003. He started working as assistant professor in USU, Logan, Utah in August 2003 and he got $70000 annually. In June 2008, he became as associated professor with $80000 annually. He got married for Eliza in April 18th, 2002. Eliza is a Mexican girl, she was born in Mexico City on Feb 2nd 1980. She came to SLC in August 2001 to get the master degree in water engineering from University of Utah. Eliza got the degree in May 2003. Eliza moved to Logan with her husband after her husband graduation. After 5 years of his marriage, he changed his name to be Joseph. Joseph and Eliza had their first baby boy "Aaron" on July 2nd 2003, and a second baby girl "Jully" on Dec 1st 2005.

A temporal model for the population consisting only of John, his wife and the two kids, between May 5th, 1976 and Dec 31st, 2016, would be as follows:

$M^R = (P, T, D, A, R, PA, PR)$, where

$P = \{p_1, p_2, p_3, p_4\}$

T = {May 5th 1976, Feb 2nd 1980, Dec 1996, Jan 1997, August 2001, April 18th 2002, May 2003, July 2nd 2003, August 2003, Dec 1st 2005, April 18th 2007, Dec 31 2016}

D = {John, Smith, Eliza, Aaron, July, Logan, Utah, SLC, White, American, Mexican, political Science, Master, Ph.D., USU, University of Utah, $80000, Male, Female}

$A = \{a_1, a_2, a_3, a_4, a_5, a_6, a_7, a_8\}$

a_1.name = FirstName, a_1.arity =4, a_1. Domain = {John, Smith, Eliza, Aaron, Jully}, a_1.roles = {Person, First Name, Start Time, End Time}, a_1.sorts = {P, d, T, T}, d ∈ a_1. Domain

a_2.name = BornIn, a_2.arity =5, a_2. Domain = {Logan, Mexico}, a_2.roles = {Person, Born City, Born Date, Start Time, End Time}, a_2.sorts = {P, d, d, T, T}, d ∈ a_2. Domain

a_3.name = GotDegreeFrom, a_3.arity =7, a_3. Domain = {Political Science, Water engineering, Master, Ph.D., University of Utah}, a_3.roles = {Person, Program name, Degree, University Nam, Graduation Date, Start Time, End Time}, a_3.sorts = {P, d, d, d, d, T, T}, d ∈ a_3. Domain

a_4.name = Race, a_4.arity =4, a_4. Domain = {White, Mexican}, a_4.roles = {Person, Race, Start Time, End Time}, a_4.sorts = {P, d, T, T}, d ∈ a_4. Domain

a_5.name = Gender, a_5.arity =4, a_5. Domain = {Male, Female}, a_5.roles = {Person, Gender, Start Time, End Time}, a_5.sorts = {P, d, T, T}, d ∈ a_5. Domain

a_6.name = IncomeAndSource, a_6.arity =5, a_6. Domain = {$80000, $110000, USU}, a_6.roles = {Person, Source, Income, Start Time, End Time}, a_6.sorts = {P, d, d, d, T, T}, d ∈ a_6. Domain

a_7.name = WorkAt, a_7.arity =4, a_7. Domain = {USU}, a_7.roles = {Person, Work Place, Start Time, End Time}, a_7.sorts = {P, d, T, T}, d ∈ a_7. Domain

a_8.name = WorkPosition, a_7.arity =4, a_8. Domain = {Assistant prof, Associated prof}, a_8.roles = {Person, Position Name, Start Time, End Time}, a_8.sorts = {P, d, T, T}, d ∈ a_8. Domain

R = {r1, r2, r3, r4}

r_1.name = MarriedTo, r_1.arity = 4, r_1.roles = {Husband, Wife, Start Time, End Time}, r_1.sorts = {P, P, T, T}

r_2.name = FatherOf, r_2.arity = 4, r_2.roles = {Father, Son/Daughter, Start Time, End Time}, r_2.sorts = {P, P, T, T}

r_3.name = MotherOf, r_3.arity = 4, r_3.roles = {Mother, Son/Daughter, Start Time, End Time}, r_3.sorts = {P, P, T, T}

r_4.name = Sibling, r_4.arity = 4, r_4.roles = {Person, Person, Start Time, End Time}, r_1.sorts = {P, P, T, T}

PA = {pa_1, pa_2, pa_3, pa_4, pa_5, pa_6, pa_7, pa_8}

pa_1.type = a_1

pa_1.data = {(p_1, John, May 5th 1976, April 18th 2007),

(p_1, Joseph, April 18th 2007, Dec 31 2016),

(p_2, Eliza, Feb 2nd 1980, Dec 31 2016),

(p_3, Aaron, July 2nd 2003, Dec 31 2016),

(p_4, Jully, Dec 1st 2005, Dec 31 2016)}

pa_2.type = a_2

pa_2.data = {(p_1, Logan, May 5th 1976, May 5th 1976, Dec 31 2016),

(p_2, Mexico, Feb 2nd 1980, Feb 2nd 1980, Dec 31 2016),

(p_3, Logan, July 2nd 2003, July 2nd 2003, Dec 31 2016),

(p_4, Logan, Dec 1st 2005, Dec 1st 2005, Dec 31 2016)}

pa_3.type = a_3

pa_3.data = {(p_1, Political Science, Ph.D., University of Utah, May 2003, Dec 31 2016),

(p_2, Water Engineering, Master, University of Utah, August 2001, Dec 31 2016)}

Pa_4.type = a_4

pa_4.data = {(p_1, White, May 5th 1976, Dec 31 2016),

(p_2, Mexican, Feb 2nd 1980, Dec 31 2016),

(p_3, White, July 2nd 2003, Dec 31 2016),

(p_4, White, Dec 1st 2005, Dec 31 2016)}

pa_5.type = a_5

pa_5.data = {(p_1, Male, May 5th 1976, Dec 31 2016),

(p_2, Female, Feb 2nd 1980, Dec 31 2016),

(p_3, Male, July 2nd 2003, Dec 31 2016),

(p_4, Female, Dec 1st 2005, Dec 31 2016)}

pa_6.type = a_6

pa_6.data = {(p1, USU, 70000, August 2003, June 2008),

(p1, USU, 80000, June, Dec 31 2016)}

pa_7.type = a_7

pa_7.data = {(p1, USU, August 2003, Dec 31 2016)}

pa_8.type = a_8

pa_8.data = {(p1, Assistant prof, August 2003, June 2008),

(p1, Associated prof, June 2008, Dec 31 2016)}

PR = {ra_1, ra_2, ra_3, ra_4}

pr_1.type = r_1

pr_1.data = {(p_1, p_2, April 18th 2002, Dec 31 2016)}

pr_2.type = r_2

pr_2.data = {(p_1, p_3, July 2nd 2003, Dec 31 2016),

(p_1, p_4, Dec 1st 2005, Dec 31 2016)}

pr_3.type = r_3

pr_3.data = {(p_2, p_3, July 2nd 2003, Dec 31 2016),

(p_2, p_4, Dec 1st 2005, Dec 31 2016)}

pr_4.type = r_4

pr_4.data = {(p_3, p_4, Dec 1st 2005, Dec 31 2016)}

4.3 Database Model (M^D)

As with real-world population, it is possible to create a temporal model that captures the evolution of a person-centric database by tracking changes to the data. As before, the model's *T* includes a special time ∞ that represent a point in the future. The model *A* and *R* would capture the database schema or at least the PII portions of the database schema. Every person represented in the database would be a person object in *P*, person attributes would be captured in *D*, and links between persons and their attributes in *PA*. The inter-person relationships would be captured in *PR*.

Whenever, a piece of data is added to the database at time t_1, then objects would be added to *P* or *D*, if they are not already there and new tuples that represent the person-to-attribute or person-to-person relationships would be added to *PA* or *PR*. The time intervals of these new tuples would be $[t_1, \infty)$. If some piece of data is delete at time t_2, then the time interval of corresponding tuples in *PA* and *PR* would be changed from $[t_1, \infty)$ to $[t_1, t_2)$. Similarly, when a piece of data is changed at t_2, the corresponding tuples in *PA* and *PR* with an interval of $[t_1, \infty)$ would be changed to $[t_1, t_2)$ and new tuples with the interval $[t_2, \infty)$ would be added.

Consider the following example where *A* includes an attribute type, a_1, where a_1.name="First Name", a_1.domain= {string}, a_1.arity = 4, a_1.roles = {Person, "First Name", Start Time, End Time}, a_1. Sorts = {P, a1.domain$_1$, T, T}. Based on the attributed type a_1, the model would include a person-attribute relation, pa_1 in *PA*. Assume that a person, p_1, named Joe was added to the database a time t_1 that a second person, p_2, was added to the database at time t_2, and p_1 changed his name to Joey at time t_3. Table 11 shows the tuples of $pa_1.data$

pa_1.type= a_1,

pa1.data = {(p_1, Joe, t_1, t_2),

(p_1, Joseph, t_2, ∞),

(p_2, Joey, t_1, ∞)}

Example 2, consider a model with a *MotherOf* relationship type, *pr* in *PR* and a mother p_1 has a child p_3 and p_4 at time t_1, at time t_2 the child p_4 has been changed his *MotherOf* relation to be a child of p_2. Table 12 shows the tuples of *pr.data*

r`.name = MotherOf, r`.arity = 4, r`.roles ={ Mother", "Child", "Start Time", "End Time"}, r`.sorts = {P, P, T, T}

Then there exists persons relation *pr* in *PR*, such that,

pr .type = r`,

pr.data = {(p1, p5, t1, ∞),

(p2, p4, t2, t3),

(p3, p4, t3, ∞)}

[t_2, ∞) would be added. If a person attribute or relationship is deleted at time t_3, then the time interval of the corresponding attribute or relation would be represented as *pa* or *pr* with an interval of [t_3, ∞).

While in a single-point timestamps Models, the time is small enough to distinguish between different events. But the smaller time unit, the more rapid the model grows. Also, as every new time point, the entire state of the population must effectively be captured in new tuples. As, the number people and attributes goes, the rate of growth goes up.

Whenever a new person is added to the population at time t_1, then person would be added to *P* and his/her attributes would be added to *D*. The relationship between the person and its attributes would be

represented by *pa* with single-point timestamp t_1, and the relationships of him/her with other existing would be represented by *pr* with the same single-point timestamp t_1. The model keep tracking of data and representing the existing *pr* and *pa* for each single-point timestamp. So, for each time t_i, a new *pa* and *pr* for the existing person *p* will be added with the single-point timestamp t_i. If a person attribute pa_x or relationship pr_x is changed at time t_2, the new person attribute or relationship *pa* or *pr* will be added to the model with a single-point timestamp t_2 and the system will stop tracking person attribute pa_x or relationship pr_x at the time-point t_2. If a person attribute or relationship is deleted at time t_3, then, also, the system will stop tracking person attribute pa_x or relationship pr_x at the time-point t_3.

Table 11. First Name

Person	Attribute Value	StartTime	EndTime
p_1	Joe	t_1	t2
p_1	Joseph	t_2	∞
P_2	Joey	t_1	∞

Table 12. Mother Relation

Mother	Child	Start Time	End Time
p_1	p_3	t_1	∞
p_1	p_4	t_1	t_2
p_2	p_4	t_2	∞

CHAPTER 5

USING PDFOL TO EXPRESS REAL-WORLD FACTS, EXPERT OPINION AND AGGREGATE KNOWLEDGE

The general idea behind determining the accuracy of the data in a person-centric database relative to the population is to compute comparable measurements from the database and the population using the same set of metrics. Ideally, data analysts would establish the metrics and apply them to models generated from the database and the population. Then, they would the compare the resulting measures, and because they were computed from the same metrics and the same kind of models (interpretations) the comparison would be meaningful.

However, as mentioned earlier, generating a model from a real-world population is impractical or prohibited. So, this research uses an alternate approach, highlighted in Figure 5-1, to guarantee that comparison of measurements is meaningful. First, data analysts define the real-world facts, expert opinions and aggregate knowledge using the real-world measurements, Y^R, and domain knowledge. In practice, Y^R comes from survey or census results and represent existing real-world measurements. For example, Y^R could include aggregated values for the ratio of baby boy births to baby girl births between 2000 and 2017. Data analyst can reverse-engineer these values into natural-language statements and then create PDFOL statements that accurately expressed them.

This reverse engineering process is sound because a real-world population can be theoretically captured in a PDFOL model and Y^R could have been computed from metrics based on the PDFOL statements. So, assuming the metrics are based on PDFOL statements that are accurately represent Y^R, then Y^D and Y^R are comparable.

This chapter explains the first part of the reverse engineering process, which is the expression of the real-world facts, expert opinions and aggregate knowledge as PDFOL statements. Chapter 6 then describes how data analysts can develop metrics based on these statements.

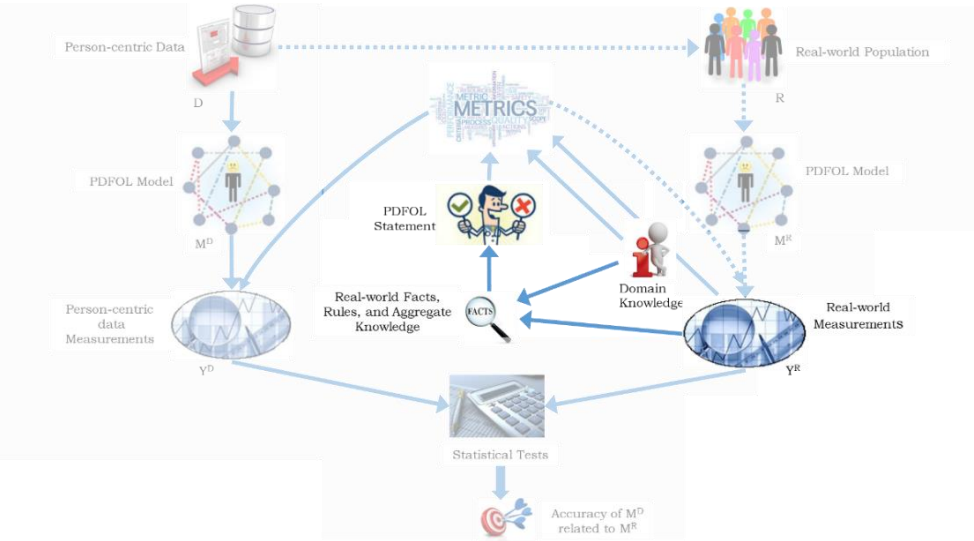


Figure 5-1. Expressing Real-world Facts, Expert Opinions and Aggregate Knowledge as PDFOL Statements

5.1. Defining and Expressing the Real-world Facts, Expert Opinions and Aggregate Knowledge

Data analyst define the real-world facts, expert opinions and aggregate knowledge using the real-world measurements, Y^R, and domain knowledge. They can reverse-engineer the values in Y^R into natural-language statements. They could write these statements as a scope and a truth – the truth should hold within the scope.

Using the standard logic terminology, the scope is a premise and the truth is a conclusion, and the two parts are connected by an implication logical connector (⇒). So, the data analyst could express the real-world facts, expert opinions and aggregate knowledge as a PDFOL statement using the format:

$$Label: Scope \Rightarrow Truth$$

Here *label* is a descriptive PDFOL statement label or name. *Scope* is a conjunction of zero or more statements that define the domain of the truth and *Truth* is conjunction of one or more of statements that represent something that should be true with the scope. For example, consider the following expert opinion "All students in a kindergarten class must be greater than or equal 5 years old". A reasonable label could be *KinderAge* and the *Scope* needs to represent persons who are students and in the kindergarten. The *Truth*

needs to say that those persons must be greater than or equal 5 years old and the whole PDFOL statement can be express as:

$$KinderAge: \forall s, t1, t2\ (\ IsStudent\ (s, t_1, t_2)$$
$$\wedge\ InClass(s, Kindergarten, t_1, t_2) \Rightarrow\ Age(s, t_1, t_2)\ \geq\ 5)$$

Both the *Scope* and the *Truth* in the real-world facts and expert opinions could be expressed as a close PDFOL statements. For example, in the above expert opinions, student *s*, time t_1 and t_2 are quantify over the for all quantifier ∀. While the aggregate knowledge could be expressed as an open PDFOL statements and their free variables are what is being constrained by it.

5.1.1. Expressing real-world facts as closed PDFOL statements

Real-world facts are hard rules about people's attributes or relations. These facts are always satisfied in real-life populations and therefore should be satisfied in person databases, violations of these facts are clear indications of incorrect or incomplete data. So, such these facts are presented by 100% of reality or truthiness in Y^R. For example, a person's death can only occur on or after a person's birth in the real-live. Therefore, in a person-centric data that contains both birth and death dates, a person's death date must be greater than or equal to that person's birth date. Violation of this constraint indicates that at least one of the dates is wrong. For example, the person gender must be either male or female and any birth date or death date cannot be on Feb. 29 for non-leap years. Below are formalizations of these sample facts as closed PDFOL statements:

$$PersonBirthDethDate: \forall p, dd, bd, t\ (\ IsPerson(p, t, t)\ \wedge DeathDate(p, \neg NULL, t, t)$$
$$\Rightarrow DeathDate(p, dd, t, t) \geq BirthDate(p, bd, t, t))$$

$$PersonGender: \forall p, t\ (IsPerson(p, t, t) \Rightarrow\ Gender(p, "Male", t, t\ ) \vee Gender(p, "\ Female", t, t\ ))$$

$$LeapBirthDate: \forall p, bd, t\ (BirthDate(p, bd, t, t)\ \wedge\ IsLeapDay(bd) \Rightarrow\ IsLeapYear(bd))$$

$$LeapDeathDate: \forall p, dd, t\ (DeathDate(p, dd, t, t)\ \wedge\ IsLeapDay(dd) \Rightarrow\ IsLeapYear(dd))$$

5.1.2. Expressing expert opinions facts as closed PDFOL statements

In this area, we look for the facts that come from the experts or discovered rules or the natural changes in a person attributes. For example, the changes in the behavior of the weight attribute during the person's life, birth time to be grown up, should match with certain kinds of curves derived from historical

health data, that means a person weight at time [t, t+1) must be within the range [$minValue_A$, $maxValue_A$], where $minValue_A$ and $maxValue_A$ are the minimum and maximum values of person weight in the real-life at the corresponding age *A* that presented by the time interval [t, t+1). These facts are satisfied by a percentage of real-life populations with very small standard deviations, and therefore should be satisfied by the same percentage in person-entities in person database. So, such these facts are presented by *r%* of reality or truthiness in Y^R. For example, person's birth date is typical more than 14 years after his/her biological mother's birth date, in the kindergarten class, the students must be greater than or equal 5 years old and the height of person must be less than or equal to 280 cm. Below are formalizations of these sample expert opinions as closed PDFOL statements:

$PersonWeight$:

$$\forall p,t,A(\ -1 \leq \min_{w||} < weight(p,w,t,t+1) \wedge\ Age(p,A,t,t) >\ -minValue_A \leq\ 1$$

$$\wedge\ -1 \leq\ \max_{w||} < weight(p,w,t,t+1) \wedge\ Age(p,A,t,t) >\ -maxValue_A \leq\ 1))$$

$MomKidAge$: $\forall m,k,t\ (IsMother(m,t,t\ )\ \wedge\ haskid(m,k,t,t\ ) \wedge$

$$Minus(Age(m,t,t\ ),Age(k,t,t\ ))\ \geq\ 14)$$

$KinderAge$: $\forall s,t1,t2(IsStudent\ (s,t1,t2) \wedge InClass(s,Kindergarten,t1,t2)$

$$\Longrightarrow Age(s,t1,t2) \geq 5)$$

$PersonHeight$: $\forall p,h,t\ (IsAdult(p,t,t\ )$

$$\Rightarrow\ Height(p,h,t,t\ )\ \leq\ 280)$$

5.1.3. Expressing aggregate knowledge as open PDFOL statements

Aggregate knowledge comes from the summation of certain data sampled from a real-world population. These facts are generally believed to be true for real-life populations with some margin of error. Individual pieces of aggregate knowledge in Y^R are typically resented by percentages. Data analysts or de-aggregation software could re-discover these facts by reverse-engineering of Y^R. For example, consider a census conducted in 2003 that reports the percentage of mothers who delivered baby boys by race. Data analysts can use this information to define an open PDFOL statement that represents the same kind of data aggregation.

Each aggregate knowledge can be expressed as an open PDFOL statement and written in the format (*Scope* ⇒ *Truth*), which we describe previously. The *Truth* has the conjunction statements that describe the knowledge that is coming from the aggregation of different knowledges. Mostly, these facts expressed by aggregate functions with free variables that is being constraints by the aggregate function. For example, assume that through census data and other studies, the percentage of mothers who delivered a baby girl in 2005 and have a white race should be some *r%* of all the new born in the same year. This could be expressed in PDFOL as follows:

$MotherWhiteRaceOf2005BabyGirl$:

$$\frac{count_{m||c,\,t1,\,t2} < MotherOf(m,c,t_1,t_2) \wedge Gender(c,"Female",t_1,t_2) \wedge BirthYear(c,2005,t_1,t_2) \wedge Race(m,\text{White},t_1,t_2) >}{count_{m||c,\,t1,\,t2} < BirthYear(c,"2005",t_1,t_2) >} \geq r\%$$

CHAPTER 6

DEVELOPING PII ACCURACY METRICS

Once the essence of existing real-world measurements, Y^R, are captured in PDFOL statements, the data analysis can follow a systematic process to formulate metrics from those statements. Figure 6-1 highlights this part of the overall process.

From each PDFOL statement *P*, data analyst can develop data accuracy metric, *ω,* to estimate one aspect the accuracy of person-centric data. In our approach, each metric *ω* has a name (*ω.name*), represented open PDFOL statement (*ω.p*), expected score (*ω.y'*) and a deviation value *(ω.d).* More formally, we define the accuracy metric as:

$$\omega : (\omega.name, \omega.p, \omega.\mathrm{y}^{\mathrm{r}}, \omega.d)$$

Where,

- *ω.name* is an identifying name or label for metric.
- *ω.p* is the PDFOL statement that express the real-world fact, expert opinion or aggregate acknowledge that metric is related to. See next paragraph for more details.

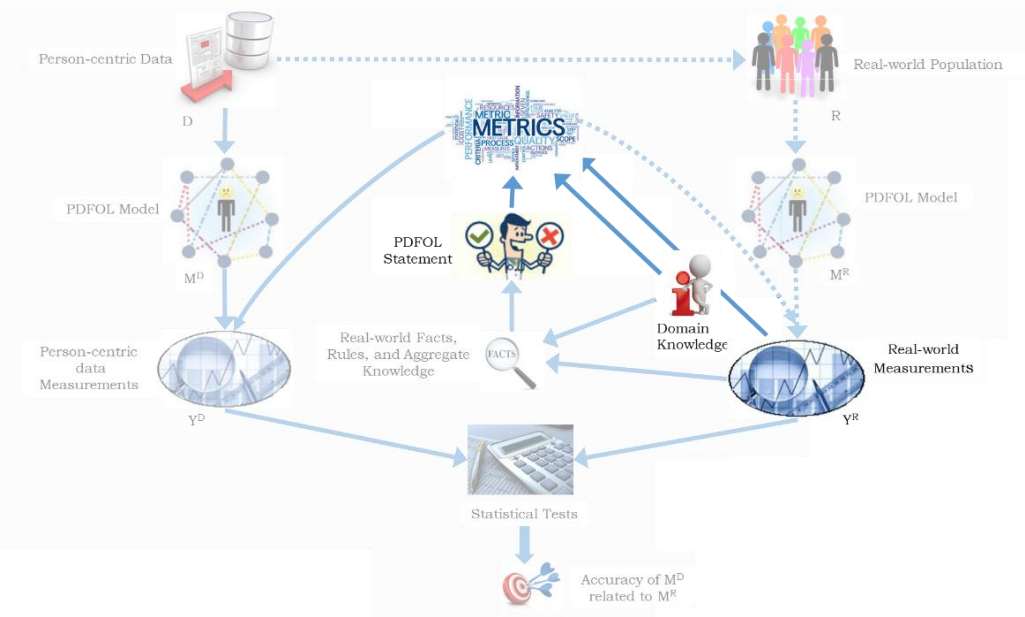


Figure 6-1. Developing PII Accuracy Metrics

- $\omega.y^r$ is a known value or set of values describe the expected score of the metric in the real-world persons, where these values defined by the real-world measurements Y^R, $\omega.y^r \in Y^R$. Any metric with null or unknown expected score, $\omega.y^r$ in Y^R, is out of our research domain.
- $\omega.d$ is a value or set of values that represent the accepted deviation of the estimated accuracy value related to this metric. In this research, the accuracy estimation gives a value between 0 and 1. 0 gives an indication of having a good accuracy value, so $\omega.d$ will be the deviation from zero. $\omega.d$ has to be defined by experts, which is presented by domain knowledge, for each metric separately.

As described in Chapter 5, each real-world fact or expert opinion is expressed as a closed PDFOL statement and each piece of the aggregated knowledge as an open statement. To create metrics, however, the data analysts must the former close statements to be open statements, so the metric effectively computes the percentage of many times those closed statements are true.

So, for a close PDFOL statement *p* that expresses a real-world fact or expert opinion, data analyst converts it to be an open PDFOL statement, by deciding what needs to be counted and computed as a percentage. In other words, certain bound variables in *p* are identified as the objects to be counted and made free variables by removing the quantifiers that bind them. Let p' be a modification of *p*, where X are the variables to be counts and the quantifiers bind X are removed. Then, PDFOL statement for the metric is as follows:

$$\omega.p = \frac{count_{X||F} < p' >}{count_{X||F} < p' > + count_{X||F} < \neg p' >}$$

In important and non-trivial part of process, is the determination of the *X* variables. The data analysts must decide what need to be counted. Typically, it is persons, like children, but it could be attributed like birth dates. It some situations, it may be valuable to count multiple things, in which case the data analysis will using the same *p* to create multiple metrics, each with a different *X* and p'.

6.1. PII Accuracy Types

PII accuracy metrics could be classified to two types based on the interpretation of it is represented PDFOL statement, $\omega.p$.

i. Singleton metric, is a developed metric with a related open PDFOL statement that has SUM, MIN, MAX, or COUNT aggregate functions with empty grouping variables *G* and returns tuple with a single value as interpretation result. For simplicity, we said it has a single numeric interpretation result. For example, a metric with a related PDFOL statement that compute the average of Assistance professor's salary at USU or the number of data tuples who are violates the expert opinions; a mom is 14 years older that her kid.

ii. Set-function metric is the metric with a PDFOL statement that yields set of tuples as a result. More precisely, the metric with an open related PDFOL statement that has at least one of the set-aggregate functions AFs with non-empty G. The related PDFOL statement in the set-function metric returns a set of tuples, each tuple with size n=|G+1|. The (n-1) values is the set of distinct values for G such that S is true for some x, and the (nth column) value has the result of applying AF on x values for which S is true for the given G. For example, find the frequency of boys and girls who born in 2003. Such these metrics, we call it also categorical metrics.

CHAPTER 7

USING PII ACCURACY METRICS TO ESTIMATE THE ACCURACY OF PERSON-CENTRIC DATA

Data analyst can apply a metric ω on the person-centric data, presented by PDFOL model M^D, to estimate one aspect of the accuracy of data. In this chapter, we present how data analysts can compute the quality assessment measurements of the person-centric data Y^D, and how they can estimate the overall inaccuracy of person-centric data using a one or more metrics.

As you see in Figure 7-1, the lighted part shows the main two steps for this part of the overall process; 1) computing the person-centric measurements Y^D and then 2) using the statistical tests to compare Y^D and Y^R with the hypothesis that they should be very similar, if D is an accurate and complete representations of R.

7.1. Compute Person-centric Measurements, Y^D

Data analyst can compute the set of quality-assessment measurement, Y^D, from a set of metrics, $\boldsymbol{\Omega}$, by applying each metric ω on M^D. Where Y^D is a set of values that represent the interpretation of the related PDFOL statement in the person-centric data, and can be defined as:

$$Y^D = \{y^d\}, \qquad y_i^d = \omega_i.p\,(M^D), \omega_i \in \Omega$$

In this way, data analyst can compute Y^D by applying the open PDFOL statements of the developed metrics on M^D. Open PDFOL statements are strongly related to queries in relational database and other database models. As mentioned previously, PDFOL aggregation functions where specifically design to parallel those found in the SQL queries. The interpretation of an open PDFOL statement is like computing the answer of a query, where the result of this interpretation is one tuple with single value or set of tuples.

7.2. Estimating the Inaccuracy of M^D Related to M^R

Data analyst can use the statistical tests to compare Y^D and Y^R. The database inaccuracy is estimated by the correlation between Y^D and Y^R. The statistical tests give a numerical value between 0 and 1, this value represents the adherence of the electronic-person data to the real-world fact, expert opinion or an aggregate knowledge related to that metric.

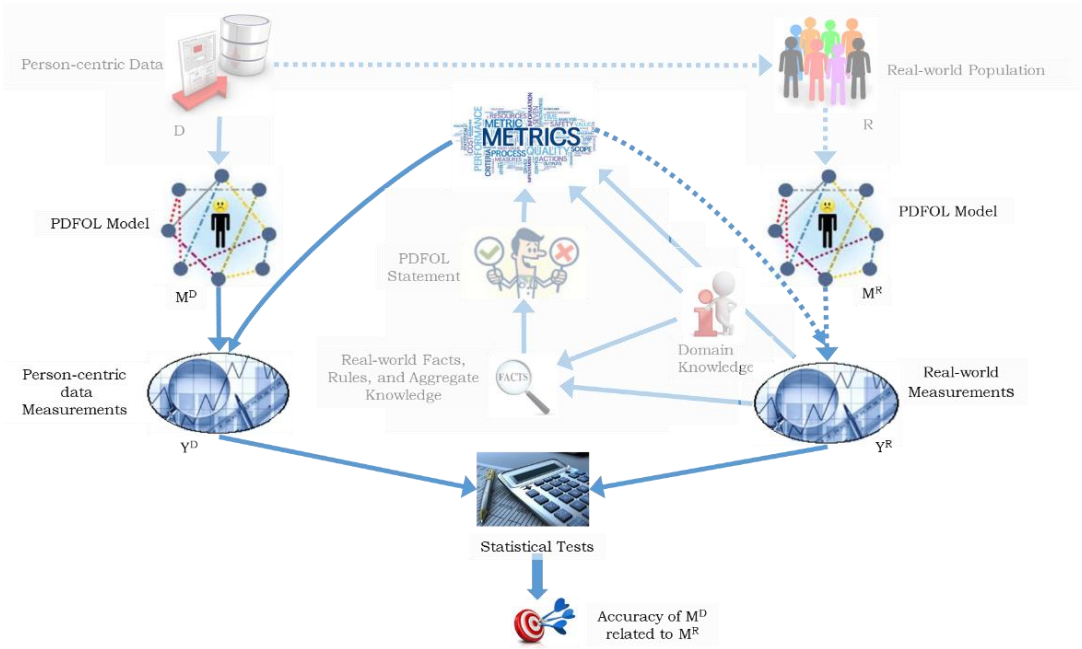


Figure 7-1. Data Accuracy Estimation

1 means no adherence to the metric and 0 means perfect adherence. The estimated inaccuracy value gives an indication whether the accuracy of data is acceptable or not, and if appropriate corrective actions should be taken. For example, if the obtained scores do not match with the expected scores, there are several possibilities; if the value is "not so bad" there is no motivation to improve the data quality. However, if the values are bad but the costs, to correct them, seems higher than the expected benefits, and then database administration may choice to do nothing. Having 1 as estimated inaccuracy value give indication of low level of accuracy and a serious corrective actions have to be taken.

To do that, data analyst have several choices of statistical test to compute the differences between the two measurements Y^D and Y^R. In this dissertation, we would use the Data Differences Test (DDT). Below, we talk about it in details and then, we present how data analyst can use this test to estimate the inaccuracy of person-centric data.

7.2.1. Data differences test

Data Differences Test (DDT), this test aims to find the degree of closeness between two values or set of values. Below, we define two instances that we were interested in this research.

1. Statistical differences between two percentages/single values

   The difference between two single values is the absolute value of the difference between the two values divided by the average of the two values. More formally, the singleton-instance of *DDT* can be defined as:

$$DDT(v_1, v_2) = \left| \frac{v_1 - v_2}{\left(\frac{v_1 + v_2}{2}\right)} \right|$$

   In terms of data accuracy, *DDT* with a two singleton values computes the differences between the person-centric measurement and the real-world measurement of a metric ω, where both values are singleton. Having 0 through the *DDT* means no difference between the values. More formally, singleton-instance *DDT* in terms of personal data accuracy can be presented as:

$$DDT\left(y_i^d, \omega_i.y^r\right) = \left| \frac{\left|y_i^d - \omega_i.y^r\right|}{\left(\frac{y_i^d + \omega_i.y^r}{2}\right)} \right|, y_i^d = \omega_i.p(M^D)$$

2. Statistical differences between two relations

   The relation is a set of tuples that is return as a resulted of the set-function metrics interpretation and it could be represented as a set of pairs (*category*, *value)*, where the *category* is a combination of *G* variables and *value* is the *AF* of *X* over distinct values of that category. The differences between relations r_1, r_2 is the average of applying the singleton-instance *DDT* on each corresponding couple of values from the two relations where these *values* can be presented as $r_1[g_{1,i} \ldots g_{n-1,i}]$, $r_2[g_{1,i} \ldots g_{n-1,i}]$. More formally, the relation-instance of *DDT* can be defined as:

$$DDT(r_1, r_2) = \frac{\sum_{i=1}^{z} DDT\left(r_1[g_{1,i} \ldots g_{n-1,i}], r_2[g_{1,i} \ldots g_{n-1,i}]\right)}{z},$$

   $where,$

   $n = \#of\ columns$

   $z = \text{number of tuples}$

In terms of data accuracy, *DDT*, with two relations y^d and $\omega.y^r$, is the average of applying the singleton-instance *DDT* on each corresponding couple of values from the two relations. More formally, relation-instance of *DDT* in terms of personal data accuracy can be presented as:

$$DDT(y^d, \omega.y^r) = \frac{\sum_{i=1}^{z} DDT\left(y^d[\mathrm{g_{1,i}} \dots \mathrm{g_{n-1,i}}],\ \omega.y^r[\mathrm{g_{1,i}} \dots \mathrm{g_{n-1,i}}]\right)}{z}$$

where,

$\mathrm{g_{1,i}} \dots \mathrm{g_{n-1,i}} \in \omega.p.G$,

$n = |G| + 1, G\ is\ the\ grouping\ variable\ in\ \omega.p$

z is the number of tuples in the relations

7.2.2. Estimating inaccuracy with singleton metric

Data analyst can use *DDT* to compute the degree of differences between the real-world measurement of a metric ω and the corresponding person-centric measurement y^d. More formally, the inaccuracy, *InAcc*, of a person-centric data *PD* related to a singleton metric ω_x can be estimated using the singleton-instance of *DDT* as:

$$InAcc(\omega_x) = DDT(y^d, \omega_x.y^r)$$

Consider, for example, a person-centric data, *Logan-PD* that represents the population of Logan city. Also, assume that census data and other studies, tells the average monthly income of employees in Logan is $4000. The data analyst define an accuracy metric $\omega_x.name$ =*EmployeeAverageIncome*, with a related PDFOL statement:

$$EmployeeAverageIncome\colon\ \frac{sum_{\mathrm{i||e,t}} < Income(e,i,t,t) >}{count_{i||e,t} < Income(e,i,t,t) >}$$

To estimate the accuracy of *Logan-PD* related to that metric. The real-world measurements $\omega_x.y^r$ for *EmployeeAverageIncome* metric in Logan city is 4000. The expert specify the deviation of *EmployeeAverageIncome*metric as $\omega_x.d$ = 0.038. Suppose the interpretation of *EmployeeAverageIncome* PDFOL statement in *Logan-PD* returns 4150, the inaccuracy of *Logan-PD* related to *EmployeeAverageIncome* metric will be:

$$InAcc(\omega_x) = \frac{|4150 - 4000|}{\left(\frac{4150 + 4000}{2}\right)} = 0.0369$$

The inaccuracy of *Logan-PD* relative to *EmployeeAverageIncome* metric is 0.0369. The value is within the accepted range, so no appropriate corrective actions need to be taken.

Example2, suppose a data accuracy metrics ω_x. With a related PDFOL predicate "*DeathNotBeforeBirth"*

$$\omega_x.p = \frac{count_{c||t,bd,dd} < birthDate(c,bd,t,t) \geq dethDate(c,dd,t,t) >}{count_{c||t} < IsPerson(c,t,t) >}$$

The data analyst can use the metrics ω_x to estimate the inaccuracy of the M^D related to that metric. Assume, *PD* has 500 people records, 3 record violate the predicate. As the related PDFOL predicate of the metric ω_x is expressed from the real-world facts then the $\omega_x.y^r$ = 100% with a zero deviation. The data analyst can compute person-centric measurement y^d as:

$$y^d = \frac{497}{500} = 0.994$$

Data analyst can compute the inaccuracy of M^D related to ω_x using the DDT as:

$$InAcc(\omega_x) = \text{DDT}(0.994,1) = 0.0012072$$

The inaccuracy of M^D relative to *DeathNotBeforeBirth* and zero deviation is 0.0012072. The value is not accepted, so appropriate corrective actions should be taken.

7.2.3. Estimating inaccuracy with set function metric

Data analyst can use DDT to compute the degree of closeness between the real-world measurement of a metric ω and the corresponding person-centric measurement y^d. More formally, the inaccuracy, *InAcc*, of a person-centric data *PD* related to a set-function metric ω_x can be estimated using the relation-instance of *DDT* as:

$$InAcc(\omega_x) = \frac{DDT(y^d, \omega_x.y^r)}{v}$$

$$v = \begin{Bmatrix} 2 & with\ metrics\ that\ compute\ the\ frequency\ or\ distribution\ of\ variables \\ 1 & others \end{Bmatrix}$$

The person-centric data has a finite number of persons, person attributes and relationships. Changing in a value of a person attribute or relationship causes to reduce the frequency of that value and by default increase the frequency of another person attribute or relation value. So, to a void duplicate counting of inaccurate data we divided the result of DDT by two.

Table 13. Born Weight Measurement in Person Data and Real-world

Born Year	Maximum Born Weight y^d	Maximum Born Weight $\omega_x.y^r$
2000	9 lbs., 4 oz.	9 lbs., 2 oz.
2001	8 lbs., 4 oz.	8 lbs., 6 oz.
2002	9 lbs., 1 oz.	9 lbs.
2003	10 lbs.	10 lbs.
2004	8 lbs., 4 oz.	8 lbs., 4 oz.

For example, suppose a metric ω_x that compute the maximum yearly weight of new baby born. Where, $\omega_x.p = \max_{w|y|c,t_1,t_2} < BornYear(c,y,t_1,t_2) \wedge BornWeight(c,w,t_1,t_2) >$. A person-centric data *Logan-Baby* has information about the new baby who delivered in Logan. Table 13 shows the born year with the two measurements; real-world measurement and person-centric data measurement.

Data analyst can measure the inaccuracy of the *Logan-Baby* data related to the metric ω_x using the data differences test with two relations as arguments as:

$$InAcc(y^d,\ \omega_x.y^r) = \frac{DDT(\{(2000,9.4),\dots,(2004,8.4)\},\{(2000,9.2),\dots,(2004,8.4)\})}{2}$$

$$= \left(\frac{DDT(9.4,9.2) + DDT(8.4,8.6) + DDT(9.1,9) + DDT(10,10) + DDT(8.4,8.4)}{10}\right) = 0.0056085$$

The estimated inaccuracy of *Logan-Baby* data related to ω_x is 0.0112085, data analyst can accept or reject the accuracy value base on the metric deviation $\omega_x.d$.

Another example, assume that through census data and other studies, it is generally believed that 45% of new baby born in Logan in 2003 are boys, 55% are girls. A person-centric data *Logan-PD* has a data about Logan citizen. *Logan-PD* has a 50000 record. The data tells there were 500 babies delivered in 2003, 230 baby boys and 270 baby girls. Data analyst can use a metric ω_x with a related predicate,

$$\omega_x.p = \frac{count_{c|g|t_1,t_2} < BirthYear(c,2003,t_1,t_2) \wedge Gender(c,g,t_1,t_2) >}{count_{c||t_1,t_2} < BirthYear(c,2003,t_1,t_2) >}$$

To estimate the inaccuracy of *Logan-PD* by applying the *DDT* is:

$$Acc(\omega_x) = DDT(y^d,\ \omega_x.y^r)/2$$

$$= DDT(\{(boy,.46),(girl,.54)\},\{(boy,.45),(girl,.55)\})/2$$

$$= \frac{|(.46 - .45)| + |.54 - .55|}{4} = .005$$

The estimated accuracy of *Logan-PD* data related to ω_x is 0.005, data analyst can accept or reject the accuracy value base on the metric deviation $\omega_x.d$.

7.2.4. Estimating inaccuracy with a set of metrics

Data analyst can estimate the inaccuracy of M^D related to a specific metric, set of metrics or the whole set of metrics in **Ω**. The estimated inaccuracy value of metric ω represent the inaccuracy of the attribute used to develop this metric. Having more and more metrics in the estimation process with good accuracy value make the M^D more trustable and close to realistic compare to M^R. Data analyst can estimate the overall inaccuracy of the M^D related to M^R using the whole set of developed metrics **Ω** and find the average of accuracy values coming from the different metrics ω in **Ω**. More formally, CAM can be defined as:

$$CAM = \frac{\sum_{i=1}^{n} Acc(\omega_i)}{n}, n = \Omega.\text{count}, \omega \in \Omega$$

CHAPTER 8

EVALUATION OF A SAMPLE ACCURACY METRIC

This chapter has the performance evaluation of a sample accuracy metrics and its predicative capability. Specifically, this chapter aims to: 1) evaluate the performance and the accuracy of a sample accuracy metric, and 2) prove that the sample metric is applicable and can be used to estimate the accuracy of person-centric data.

8.1 Evaluating the Developed Accuracy Metrics

I evaluate the performance of a sample accuracy metric by studying its predictive capabilities. Specifically, I generated a synthetic population of 10300 persons that included race information in the year 2003. I assume the synthetic data represent a real-world population M^R and I develop a sample accuracy metric, ω, for aggregate knowledge about the "*frequency of person's race in 2003*". The PDFOL statement for ω is as follows:

$$\omega.p = \frac{count_{x|r|} < race(x, r, 2003{,}2003) >}{count_{x|\ \ |t1,t2}(IsLiveperson(x, 2003, t1, t2))}$$

Then, I run three performance tests on the data set using the following steps:

a. Mutate the synthetic population test data in a specific way to generate a new version(M_i^D). We build the new version by resampling the observed generated data. Specifically, we shuffle the testing data M^R by assigning different outcome values to some known observation from among the set of actually observed outcomes.

b. Compute the real accuracy of the (M_i^D) directly by comparing the race attribute with the corresponding represented real-world data which is presented by the synthetic population test data(M^R).

c. Use the DDT to estimate the accuracy of the (M_i^D) related to the developed metric

d. Compare the values of steps b and c

e. Repeated steps 10 times until we get conclusions about the performance of the metrics can be rigorously established.

The sample metric estimate the accuracy of the data by computing the average differences in distribution over a grouping. Where any change in person attribute "race" in the generated data

will change the distribution of the attribute values. The evaluation cover the different possible changes in data by having three tests.

Test I- Evaluation with small changes in data

With this test, I mutate the synthetic population test data with small number of changes; starting with a small number of inaccurate person records, 50 persons, which represent 0.5% of data. For each iteration of the test, I increase the number of the persons who have inaccurate attribute race for more 50 persons. Table 14 shows the number of changes, the computed true real inaccuracy and the estimated inaccuracy using the developed performance and the DDT. The result obtained from the test are summarized by Figure 8.1. The figure shows how the real inaccuracy and estimated inaccuracy are very close and correlated to each other with very small deviations. When the noisy data increase the real inaccuracy and estimated inaccuracy increase.

Test II- Evaluation with middle changes in data

Through test II, I mutate the synthetic population test data with a middle number of changes starting with a small number of inaccurate person records, 500 persons, which represent 5% of data. For each iteration of the test, I increase the number of the persons who have inaccurate attribute race for more 500 persons until we get a 50% of inaccurate data.

Table 14. Test I Data Changes Number and Inaccuracy Results

Number of changes	Real Inaccuracy	Estimated Inaccuracy
0	0	0
50	0.00484543	0.004858505
100	0.00969086	0.00965421
150	0.01453629	0.014514305
200	0.01938172	0.019337105
250	0.02422715	0.02420628
300	0.02907258	0.02927306
350	0.03391802	0.034126005
400	0.03876345	0.03928838
450	0.04360888	0.044311815
500	0.04845431	0.04930476

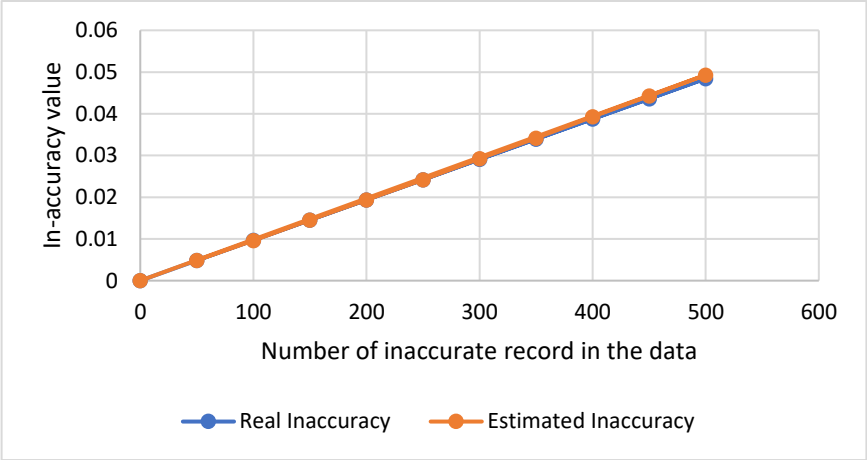


Figure 8-1.Test-I Results

Table 15 shows the number of inaccurate record in the data, the computed real inaccuracy and the estimated inaccuracy using the developed performance and the DDT. The result obtained from the test are summarized by Figure 8-2. The figure shows how the real inaccuracy and estimated inaccuracy are close and correlated to each other with small deviations. The estimated inaccuracy value starts at point 1500 persons with inaccurate data to be little bit higher than the real inaccuracy. These differences are already covered by having a deviation value for each metric separately. So, based on that the estimated accuracy still give a good representation of the real inaccuracy. To see the behavior and the deviation between the real-inaccuracy and estimated inaccuracy, see test three.

Test III- Evaluation with significant changes in data

Through test III, I mutate the synthetic population test data with a significant number of changes. In this test, I mutate the testing data in different way. Randomly I choose a value from the domain of the person attribute race. Then I replace it with other values from the domain, where these values having frequency in the testing data. I repeat this step 10 times until I get testing data with same number of values in the person attribute race domain in the year 2003. Using this mutation give a chance to have a testing data with very high percentage of inaccurate. Table 16 shows the number of inaccurate record in the data, the computed real inaccuracy and the estimated inaccuracy using the developed performance and the DDT.

Table 15. Test II Data Changes Number and Inaccuracy Results

Number of changes	Real In-accuracy	Estimated In-accuracy
0	0	0
500	0.04845431	0.04930476
1000	0.09690862	0.0992536
1500	0.14536292	0.15113084
2000	0.19381723	0.2059563
2500	0.24227154	0.26495107
3000	0.29072585	0.32815697
3500	0.33918015	0.39976739
4000	0.38763446	0.48097801
4500	0.43608877	0.57685448
5000	0.48454308	0.60192134

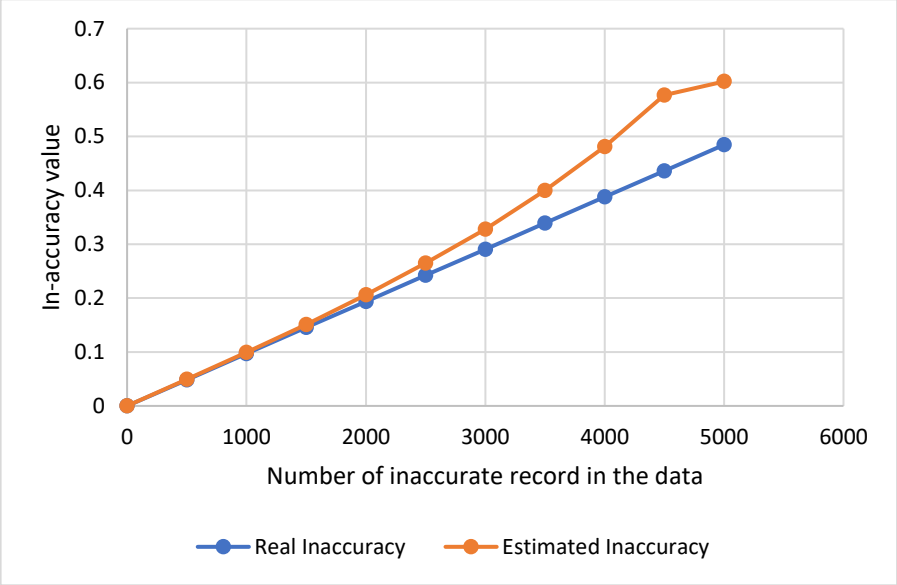


Figure 8-2. Test-II Results

The result obtained from the test are summarized by Figure 8-3. The figure shows how the real-accuracy and estimated accuracy almost start and end in the same points. Where the point of small number of changes represents the accuracy of data with small inaccurate data and the other point represent the data with high percentage of inaccurate data. From the Figure 8.3 also, we can see, the biggest differences between the real inaccuracy and the estimated inaccuracy is around

having a 50% of inaccurate data. Having a deviation of each metrics eliminate this shifting in the values and keeping the estimated value within the bounded channel of the real inaccuracy.

Table 16. Test III Data Changes Number and Inaccuracy Results

Number of changes	Real-Accuracy	Estimated Accuracy
0	0	0
915	0.088671	0.133096
1863	0.180541	0.262548
2826	0.273864	0.387882
3281	0.352076	0.505925
4702	0.455664	0.615449
5636	0.546177	0.716008
6574	0.637077	0.806181
7525	0.729237	0.883136
8460	0.819847	0.944467
9383	0.909294	0.984879

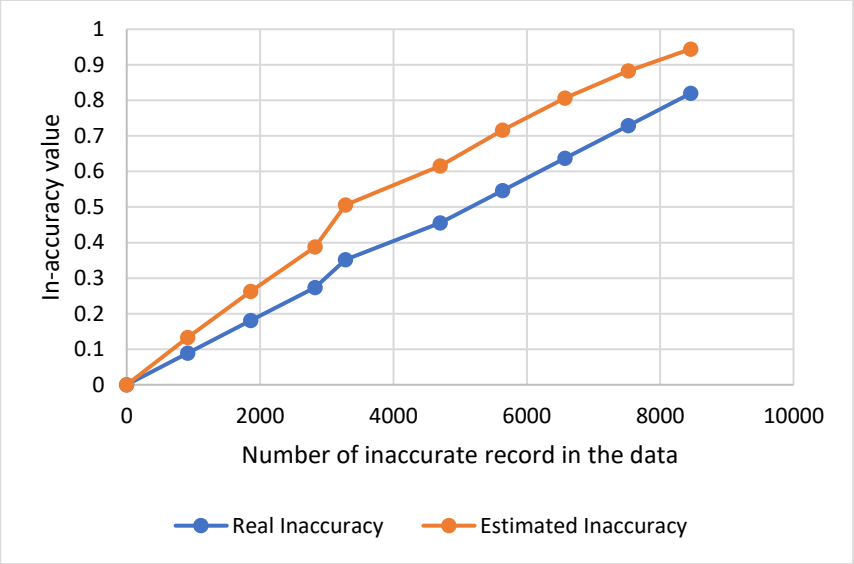


Figure 8-3. Test-III Results

8.2 Evaluating the Usability and Applicability of the Sample Accuracy Metrics

In accomplishing this goal, we develop a testing tool related to our own test data and a sample accuracy metrics. Below, we give more details on each.

1. Creating test data

   I use Postgres 9.6 to create a test data, which we called *Person-Centric Database*. As you can see in Figure 8-4, the database has three tables; *PersonRecord*, *Race* and *FamilyRelationship*. *PersonRecord* table represents ten of the real-world person properties, and the eleventh property is captured by the *Race* table. *FamilyRelationship* table represents the inter-person relationships *motherOf* or *fatherOf*.

   I generate the data test in a way that is covers some of the most interesting situations found in real populations. It represents an interesting individuals, characteristics, and relationships, which make it diverse to argue and can be used to create different types of metrics; singleton and set-function metrics. Our test data is synthetic population with a diverse set of individuals. We believe our test data is sufficiently realistic and large; it represents 10000 individuals in the real-world. In addition, we keep the generated data 100% accurate compare to the real-world facts

   I suppose that the generated synthetic population test data represents the real life-world in the metrics evaluation step and we refer it by D^R. To keep creating test data process simple, we use the generated data to compute the expected values of metrics that are related to the expert opinions, and aggregate statistics constraints.

2. Developing data accuracy metrics

   In this step, I develop the set of metrics based on the person's attributes and relationships that are represented by the testing data set. Below, I give the metrics in the natural language:

   - person's birth date is typical more than 14 years after his/her biological mother's birth date
   - The percentage of new baby boys in the year y, and the percentage of new baby girls in the same year y is

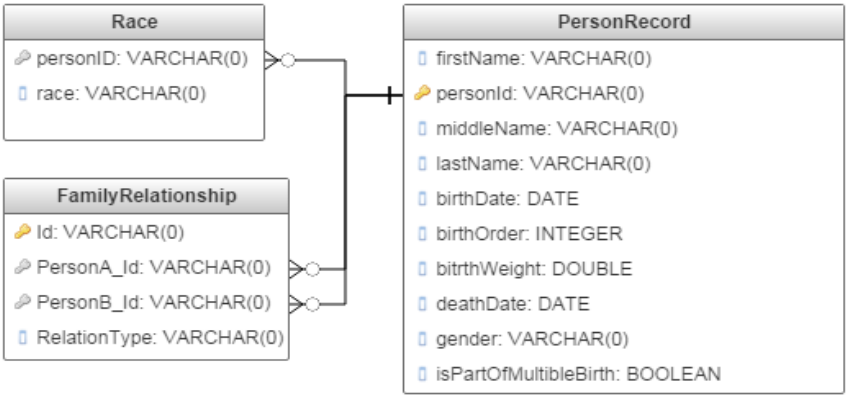


Figure 8-4. Testing Data Schema

- The frequency of new baby names $\{m_1, m_2, \ldots, m_k\}$ in the year y is $\{n_1, n_2, \ldots, n_k\}$ respectively
- The frequency of new baby born in the year y and with born weight w
- The frequency of mothers who had a baby boy in the year y and have race r
- The frequency of mothers who had a baby girl in the year y and have race r
- The frequency of mothers who had a baby in the year y and have race r

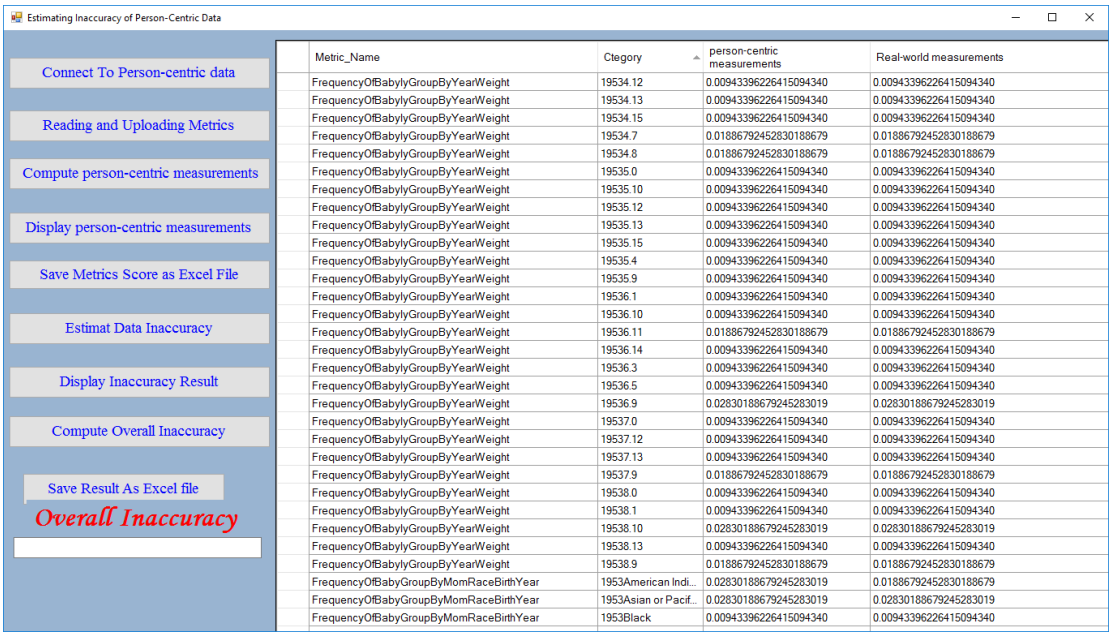
3. Testing tool

I use the .NET framework, C# language to build the testing tool, which called data accuracy estimation tool. Testing tool has two inputs: person-centric data and the developed accuracy metrics as xml-file. As you can see in Figure 8.5, the tool can compute the person-centric measurements Y^D, compute the estimated inaccuracy for each metric and compute the overall accuracy of the person-centric data related to the metrics defined on the input.xml file. The tool allow the user to save the result as an excel file.

4. Testing the testing tool

I assume the generated test data represent the real-world population M^R. To test the testing tool, I apply the defined metrics on the test data to compute the real-world measurements Y^R. Then, I estimate the inaccuracy of testing data related to the defined metrics and the computed Y^R. The tool

gives zero value as a result on inaccuracy estimation for each metric individually and a zero for the overall inaccuracy, which give some grantee that the tool is working in correct way.



Metric_Name	Ctegory	person-centric measurements	Real-world measurements
FrequencyOfBabylyGroupByYearWeight	19534.12	0.00943396226415094340	0.00943396226415094340
FrequencyOfBabylyGroupByYearWeight	19534.13	0.00943396226415094340	0.00943396226415094340
FrequencyOfBabylyGroupByYearWeight	19534.15	0.00943396226415094340	0.00943396226415094340
FrequencyOfBabylyGroupByYearWeight	19534.7	0.01886792452830188679	0.01886792452830188679
FrequencyOfBabylyGroupByYearWeight	19534.8	0.01886792452830188679	0.01886792452830188679
FrequencyOfBabylyGroupByYearWeight	19535.0	0.00943396226415094340	0.00943396226415094340
FrequencyOfBabylyGroupByYearWeight	19535.10	0.00943396226415094340	0.00943396226415094340
FrequencyOfBabylyGroupByYearWeight	19535.12	0.00943396226415094340	0.00943396226415094340
FrequencyOfBabylyGroupByYearWeight	19535.13	0.00943396226415094340	0.00943396226415094340
FrequencyOfBabylyGroupByYearWeight	19535.15	0.00943396226415094340	0.00943396226415094340
FrequencyOfBabylyGroupByYearWeight	19535.4	0.00943396226415094340	0.00943396226415094340
FrequencyOfBabylyGroupByYearWeight	19535.9	0.00943396226415094340	0.00943396226415094340
FrequencyOfBabylyGroupByYearWeight	19536.1	0.00943396226415094340	0.00943396226415094340
FrequencyOfBabylyGroupByYearWeight	19536.10	0.00943396226415094340	0.00943396226415094340
FrequencyOfBabylyGroupByYearWeight	19536.11	0.01886792452830188679	0.01886792452830188679
FrequencyOfBabylyGroupByYearWeight	19536.14	0.00943396226415094340	0.00943396226415094340
FrequencyOfBabylyGroupByYearWeight	19536.3	0.00943396226415094340	0.00943396226415094340
FrequencyOfBabylyGroupByYearWeight	19536.5	0.00943396226415094340	0.00943396226415094340
FrequencyOfBabylyGroupByYearWeight	19536.9	0.02830188679245283019	0.02830188679245283019
FrequencyOfBabylyGroupByYearWeight	19537.0	0.00943396226415094340	0.00943396226415094340
FrequencyOfBabylyGroupByYearWeight	19537.12	0.00943396226415094340	0.00943396226415094340
FrequencyOfBabylyGroupByYearWeight	19537.13	0.00943396226415094340	0.00943396226415094340
FrequencyOfBabylyGroupByYearWeight	19537.9	0.01886792452830188679	0.01886792452830188679
FrequencyOfBabylyGroupByYearWeight	19538.0	0.00943396226415094340	0.00943396226415094340
FrequencyOfBabylyGroupByYearWeight	19538.1	0.00943396226415094340	0.00943396226415094340
FrequencyOfBabylyGroupByYearWeight	19538.10	0.02830188679245283019	0.02830188679245283019
FrequencyOfBabylyGroupByYearWeight	19538.13	0.00943396226415094340	0.00943396226415094340
FrequencyOfBabylyGroupByYearWeight	19538.9	0.01886792452830188679	0.01886792452830188679
FrequencyOfBabyGroupByMomRaceBirthYear	1953American Indi...	0.02830188679245283019	0.01886792452830188679
FrequencyOfBabyGroupByMomRaceBirthYear	1953Asian or Pacif...	0.02830188679245283019	0.02830188679245283019
FrequencyOfBabyGroupByMomRaceBirthYear	1953Black	0.00943396226415094340	0.00943396226415094340

Figure 8-5. Data Accuracy Estimation Tool

CHAPTER 9

RELATED WORK

Data quality is an issue for many different disciplines, such as statistics, management, and computer science. Existing research results show that data quality researches primarily operate in two major disciplines: Management Information System (MIS) and Computer Science (CS) [23]. In the beginning of 1980, researchers start focusing on how to control data manufacturing and how to detect the data quality problems. In 1990, computer scientists identify the problems by defining, measuring, and improving the quality of electronic data in data bases, data warehouses, and legacy systems [23].

In this section, I present research related to the accuracy of integrated data and discuss the prior work that supports the quality of integrated data and the measurement the accuracy of data.

9.1. Works in Data Integration

Several researches have been done to develop technologies that assessing, improving and managing data quality in integrated databases [24]. Researchers develop different techniques to design systems that can produce data with high quality in different quality dimensions; consistency, completeness, accuracy, accessibility, and other dimensions [1] [25] [26] [27] [28] [29] [30] [31]. Some researchers describe the problem and focus on how to characterize it [23] [24]. M. Gretz suggests a new taxonomy for data quality in the integrated data system. The new taxonomy has the most important data quality aspects including data accuracy, which are modeled as a metadata [23]. Other researchers have tried to solve schema and semantic heterogeneity problems to increase the quality and accuracy of data in the integrated systems. Some researchers try to solve the schema heterogeneity by using different techniques [25][32][33]. C. Batini, et al., [26] construct a global schema for data warehousing. While others try to develop new match algorithms, and implementing the schema matching components [25][34].

Researchers try to discover and solve the semantic conflicts among heterogeneous systems [28-35]. C. Batini, et al., [26] represent data semantic in disparate systems by using a logic-based object-oriented framework. P. Vassiliadis, et al., [36] develop a new tool to solve the semantic problems by providing a

uniform Meta model. W. Fan, et al. [30] classify the data value conflicts into two categories: context independent and context dependent. The context independent conflicts are caused by unexpected error, while the dependent conflicts are the result of heterogeneity of data sources. And then they propose some conversion rules to describe the quantitative relationships among data values involving context dependent conflicts. X. Xu, et al., [31] concern with how to solve problems of heterogeneity in biological knowledge integrated data by presenting a way to classify live objects into different categories. C. E. Varghese and G. N. Sundar, [35] have demonstrated a new way to use the Context Interchange (COIN) technology to capture data semantics and reconcile semantic heterogeneities.

To support the quality of integrated data; many techniques and methods also are used to detect the duplicate records. Ektefa [37] talks about some of these methods such as Decision Tree, Nave Bayes and Bayesian Networks. He did an experiment to find the effectiveness of the classifier in these methods. He finds that the effectiveness of the classifier depends on the input dataset. Researches on the topic of record linkage go into three different areas; (1) Researchers present an online record matching methods to address and solve the problems of record matching in the web database query results [35] [38]. (2) Researchers try to find the matching records in the integrated database [1] [25] [39]. And (3) Researchers review and validate the record linkage procedures by using different methods and techniques such as probabilistic measurements [40].

However, different works have been done to support data integration and its process, there is no guaranty that the generated data has a 100% accuracy. So, researchers propose and develop different methods and techniques to measure the quality of integrated data and its accuracy.

9.2. Works in Data Quality Measurements

Measuring accuracy of data is not simple. So, researchers propose and develop different methods and metrics to measure it. Below, I give some of these relevant works.

M. Bobrowski et al. [41] propose a metric to be used as the start point for systematic analysis of data quality. They use the traditional software metric technique and following the Goal-Question-Metric (GQM) to generate the metrics that measure both the set of data and the data model. A. Doan and A. Y.

Halevy, [24] provide a description for both the subjective and objective assessments of the data quality and they develop an objective data quality metrics.

Many researches have been proposed in the heterogeneous database area [42] [43]. B. Carlo et al., [43] present a methodology for data quality assessment and improvement that called Heterogeneous Data Quality Methodology (HDQM). In their research, they deal with three types of data: structured data represented in databases, semi structured data usually represented in XML, and unstructured data represented in documents. They practice the methodology using the accuracy and currency quality dimensions. B. Piprani and D. Ernst, [44] add the idea of using the score-card approach to rate and assess the quality of data. Their new model allows the data to pass through a data-quality filter and some data-quality firewalls by establishing sluice gate parameters.

Some researchers prefer to propose probability-based metrics to measure the quality [45]. While some researchers propose accuracy metrics based on the type of data being measured. For example, [46] [30] propose metrics to measure the accuracy of continues historical variables that are changed periodically over time and can take any value such as the height of a person, which can take any value, such as 141.35 cm, 127.371 cm, and so on. Or some researchers estimate the accuracy of data by using the statistical methods to compute the closeness of the analytical result to the true value [45] [47], or to build a data quality rules [48]. Researchers propose different way to find the business rules and using them to measure the quality of data such as the work presented by [37] [49] [43] [50] and others.

V. Chanana and A. Koronios, [51] present a framework to build and develop a business application by using external business logic. In their work, they separate the business rules from the application. They store the business rules in a centralized repository. Decoupling the business rules from the application can provide many benefits that make the application maintainable, extendable and modifiable. Also, it makes the rule repository reusable. They also present and describe the data quality rule classes and the rules of each class. Their approach can help the developed application to be more agile. F. Chiang and R. J. Miller, [52] define the quality rules using a relational expression. H. Ruan et al., [53] introduce a novel integrated platform for the data quality analysis based on the technique of meta-model and regular expressions that represent the data quality evaluation rules.

Other researchers use data mining methods and techniques to discover the data quality rules [50] [54] [55] [56]. F. G. Alizamini, et al., [50], researchers propose a new method to measure the accuracy dimension of data quality using fuzzy association rules. The new method is presented to improve the data quality by discovering the hidden rules in the datasets, they use the user knowledge and background about data in the process of determining the accuracy and ultimately total quality of dataset. M. S. Shahriar and S. Anam, [54], the researchers propose a new data quality framework that is towards the broader task of data mining and data quality for XML data integrations. While proposing, they consider XML constraints to be used for XML data quality measurements. They used these constraints to find patterns and association rules in XML data mining. The constraints, which they are imposing, play an important role for data quality; they help to improve data quality of XML and to get efficient data mining in XML. O.H. Choi et al., [55] propose an efficient methodology that assuring the quality of a data with considering a dynamic clustering method in heterogeneous environments. They use ontology, which consist of meaningful words, to extract and evaluate data quality rules. They extract the relationship through ontology words, and then generate SQL to evaluate the unexpected business rules. J. Hipp et al., [56] introduce a new promising Data Quality Mining (DQM) approach based on the academic and the business point of view. DQM supports the data quality measurement and improvement. The goal of it is to detect, quantify, explain, and correct data quality deficiencies in very large databases. They also describe how to employ association rule mining for the purpose DQM. They provide a typical application scenario of their DQM to support the knowledge discovery in databases (KDD) projects, especially during the initial phases.

Researchers use the functional dependencies to define the quality rules [52] [57]. F. Chiang and R. J. Miller, [52] propose a new data driven tool with effective algorithm. The tool discovers and searches for minimal conditional functional dependencies (CFDs) and dirty values which are hold over a given data instance. The tool used to suggest the possible rules within an organization's data quality management process. They use the rules to identify conformant and non-conformant records. P. Z. Yeh and C. A. Puri, [57] propose an approach that discovers effective conditional Functional Dependencies (CFDs) for detecting inconsistencies in data. The approach improves data quality by detecting the inconsistency in the data. They describe the process of generating the candidate CFD, and how to refine each candidate CFD by comparing it with records from the relation of interest. They evaluate their approach on three real world data sets.

CHAPTER 10

CONCLUSION AND FUTURE WORK

The benefits of integrated person data can only be realized if the data are accurate, relatively complete, timely, and pertinent. Without a valid assessment of accuracy there is a risk of data users coming to incorrect conclusions or making bad decision based on inaccurate data. The most direct way to measure data accuracy of PII would be to compare the data with real-world people, individual by individual, attribute by attribute, relationship by relationship. This is impractical because the labor-intensive nature such a comparison, and it is impossible for many person databases because of the confidential nature of data and the inaccessibility of the real individuals. So, the problem of estimating the person data accuracy becomes one of estimating data accuracy using real-world facts, expert opinions, or aggregate knowledge about the represented population

To address the problem of estimating data accuracy using real-world constraints, expert opinions, and aggregate knowledge, this research has address five important sub-problems: 1) the development or adaptation of a formalism for modeling and reasoning about real-world and electronic person data, their attributes, and relationships, 2) the development of methods for expressing real-world facts, expert opinions, and aggregation knowledge using this formalisms, 3) the development a method that can help data analyst to develop metrics that can estimate a database accuracy based on the real-world facts, expert opinions, and aggregation knowledge and the real-world and person-centric data formalisms and models, 4) the application of accuracy metric to person-centric data to compute the quality assessment measurements and 5) comparing the person-centric measurements with the real-world measurements.

This research proposed an extended first-order logic language (FOL), called PDFOL (Person Data First-order Logic). The language has salient features that give it the power to express relevant person attributes, inter-person relations and the different kinds of facts and rules, and to model person-centric databases, enabling formal and efficient reason about their accuracy and provide a foundation for methods for reasoning about the accuracy of PII.

This research introduced a model structure for PDFOL and the rules for mapping the language's symbols to objects and relations defined by the structure. In other words, I established a theoretical foundation for PDFOL's semantics that described how PDFOL model can be automatically generated from person-centric data with minimal the number of tuples necessary to accurately and completely capture all the additions, changes, or deletions in the person-centric data.

With its syntax and semantics formalized, PDFOL provided a mechanism for expressing data-accuracy metrics, computing measurements using these metrics on person-centric databases, and comparing those measurements with expected values from real-world populations. In details, data analysts formalize the facts and expert opinions by expressing them as closed PDFOL statements and the aggregate knowledge as open PDFOL statements. Then, they use these PDFOL statements to develop metrics. After that, data analysts apply these metrics to M^D to compute the quality-assessment measurements, Y^D. Finally, they use statistical tests to compare Y^D with the real-world measurements, Y^R. The database inaccuracy is estimated by the correlation (or lack thereof) between Y^D and Y^R.

I evaluated the performance of a sample accuracy metric and its predicative capability through data-mutation evaluation process. The process included three tests that covered the different amount of inaccurate data: small percentage, almost 50%, and up to 90%. The three tests show that the estimated inaccuracy is approximately the same as the real inaccuracy. Also, I showed that the sample metric is applicable and can be used to estimate the inaccuracy of person-centric data. I developed and implemented a data accuracy estimation tool relative to specific person-centric data and accuracy metrics.

Future work includes developing a suite of reusable metrics that will apply to many common populations and validate those metrics and their quality-assessment capabilities using data-mutation testing technique. These metrics will be presented in PDOFL and based on the formalisms presented in this research. Also, I plan on doing a deep study and data-mutation testing technique to 1) discover the relation between the metrics that have common person attributes or relationships in their definitions, 2) find a methodology that help data analysts to develop a comprehensive accuracy metric related to their own data and metrics, where each metric can have a weight.

Another future topic is to provide a foundation for methods of reasoning about other data qualities, like timeliness and consistency.

REFERENCES

[1] S.Jain, "A Merging System for Integrated Person-Centric Information Systems," Utah State University, 2011.

[2] "Interpretation (logic)," Wikipedia, 11 11 2016. [Online]. Available: https://en.wikipedia.org/wiki/Interpretation_(logic). [Accessed 18 5 2017].

[3] H.D. Ebbinghaus, J. Flum and W. Thomas, Mathematical Logic, New York: Springer, 1994.

[4] P. B. Andrews, An Introduction to Mathematical Logic and Type Theory: To Truth Through Proof (Applied Logic Series), Springer, 2001.

[5] E. Mendelson, Introduction to Mathematical Logic, CRC Press, 2009.

[6] R. A. Freire, "First-Order Logic and First-Order Functions," *Springer Basel*, 2015.

[7] "Wikipedia, the free encyclopedia," 2016. [Online]. Available: https://en.wikipedia.org/wiki/Functional_completeness. [Accessed 28 11 2016].

[8] M. Team, "WolframMathWorld," Mathmatica Technology, 9 1 2017. [Online]. Available: http://mathworld.wolfram.com/Interpretation.html. [Accessed 11 1 12017].

[9] I. Pratt-Hartmann, "Logics with Counting," School of Computer Science, Manchester, UK.

[10] A. Natsev, G. Fuh, W. Chen, C.-H. Chiu and J. Vitter, "Aggregate Predicate Support in DBMS," in *Thirteenth Australasian Database Conference (ADC2002)*, Melbourne, Australia, 2002.

[11] "wikipedia," wikipedia, 3 12 2016. [Online]. Available: https://en.wikipedia.org/wiki/Substitution_(logic). [Accessed 12 2 2017].

[12] I. Partt-Hartmann, "Logics with counting," University of Manchester M13 9PL, UK, Manchester.

[13] I. Glöckner, Fuzzy Quantifiers: A Computational Theory, Springer, 2008.

[14] S. Abitebou, L. H. and J. V. d. Bussche, "Temporal Versus First-Order Logic to Query Temporal Databases," in *ACM Symposium on Principles of Database Systems*, 1996.

[15] J. Chomicki and D. Toman, "Temporal Logic in Database Query Languages," University at Buffalo USA,University of Waterloo Canada, 2005.

[16] P. Wolper, "Temporal logic can be more expressive," *Information and Control*, 1983.

[17] "Backus–Naur form," Wikipedia, 2 5 2017. [Online]. Available: https://en.wikipedia.org/wiki/Backus%E2%80%93Naur_form. [Accessed 10 5 2017].

[18] S. W. Clyde, D. W. Embley, S. W. Liddle and S. N. Woodfield, "OSM-Logic: A Fact-Oriented, Time-Dependent Formalization of Object-oriented Systems Modeling," *Springer Link*, vol. 7260, pp. 151-172, 2012.

[19] L. Liu and M. Tammer, "Two-Sorted First-Order Logic," in *Encyclopedia of database systems*, Springer US, 2009, p. 3220.

[20] P. Ferraris and V. Lifschitz, "On the Stable Model Semantics of First-Order Formulas with Aggregates," Google, Inc.

[21] "wikipedia," 5 1 2017. [Online]. Available: https://en.wikipedia.org/wiki/Projection_(relational_algebra). [Accessed 12 2 2017].

[22] Appcelerator, "Axway Appcelerator Blog," 1 7 2010. [Online]. Available: http://www.appcelerator.com/blog/2010/07/how-to-perform-crud-operations-on-a-local-database/. [Accessed 20 4 2017].

[23] M. Gertz, "Managing data quality and integrity in federated databases," in *Integrity and Internal Control in Information Systems*, Springer, 1998, pp. 211-229.

[24] A. Doan and A. Y. Halevy, "Semantic integration research in the database community: A brief survey," *AI magazine,* 2005.

[25] "Managing Schematic Heterogeneity in Database Management Systems," [Online]. Available: http://www09.sigmod.org/disc/disc99/disc/nsf_idm/Imported/rjmiller.html. [Accessed: 25-Mar-2016]..

[26] C. Batini, M. Lenzerini and S. B. Navathe, "A comparative analysis of methodologies for database schema integration," *ACM computing surveys (CSUR),* pp. 323-364, 1986.

[27] E. Rahm and P. A. Bernstein, "On matching schemas automatically," *VLDB Journal,* pp. 334-350, 2001.

[28] C. H. Goh, S. Bressan, S. Madnick and M. Siegel, "Context interchange: New features and formalisms for the intelligent integration of information," *ACM Transactions on Information Systems (TOIS),* p. 270–293, 1999.

[29] P. Vassiliadis, Z. Vagena, S. Skiadopoulos, N. Karayannidis and T. Sellis, "ARKTOS: towards the modeling, design, control and execution of ETL processes," *Information Systems,* p. 537–561, 2001.

[30] W. Fan, H. Lu, S. E. Madnick and D. Cheung, "Discovering and reconciling value conflicts for numerical data integration," *Information systems,* p. 635–656, 2001.

[31] X. Xu, A. C. Jones, W. A. Gray, N. J. Fiddian, R. J. White and F. A. Bisby, "Design and performance evaluation of a web-based multi-tier federated system for a catalogue of life," in *Proceedings of the 4th international workshop on Web information and data management*, 2002.

[32] S. Cluet, P. Mogilevsky, J. Siméon, S. Zohar, P. A. Bernstein, T. Bergstraesser, M. Carrer and A. Joshi, "Data Engineerng".

[33] E. Rahm and H. H. Do, "Data cleaning: Problems and current approaches," *IEEE Data Eng. Bull.,* pp. 3-13, 2000.

[34] S. Chawathe, H. Garcia-Molina, J. Hammer, K. Ireland, Y. Papakonstantinou, J. Ullman and J. Widom, "The TSIMMIS project: Integration of heterogenous information sources," 1994.

[35] C. E. Varghese and G. N. Sundar, "Record Matching : Improving Performance in Classification," *Academic Journal*, p. 1207, 2011.

[36] P. Vassiliadis, Z. Vagena, S. Skiadopoulos, N. Karayannidis and T. Sellis, "ARKTOS: towards the modeling, design, control and execution of ETL processes," *Information Systems*, p. 537–561, 2001.

[37] Ektefa, "A Comparative Study in Classification Techniques for Unsupervised Record Linkage Model," vol. 7, no. 3, 2011.

[38] W. Su, J. Wang and F. H. Lochovsky, "Record Matching over Query Results from Multiple Web Databases," vol. 22, no. 4, 2010.

[39] P. Christen, "Automatic record linkage using seeded nearest neighbour and support vector machine classification," in *Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining*, 2008.

[40] D. P. d. Silveira and E. Artmann, "Accuracy of probabilistic record linkage applied to health databases: systematic review," vol. 43, no. 5, 2009.

[41] M. Bobrowski, M. Marré and D. Yankelevich, "Measuring data quality," 1999.

[42] B. T. Dai, N. Koudas, B. C. Ooi, D. Srivastava and S. Venkatasubramanian, "Column heterogeneity as a measure of data quality.," 2006.

[43] B. Carlo, B. Daniele, C. Federico and G. Simone, "A Data Quality Methodology for Heterogeneous Data," vol. 3, no. 1, 2011.

[44] B. Piprani and D. Ernst, "A model for data quality assessment," in *On the Move to Meaningful Internet Systems: OTM 2008 Workshops*, 2008.

[45] B. Heinrich, M. Klier and M. Kaiser, "A Procedure to Develop Metrics for Currency and its Application in CRM," vol. 1, no. 1, 2009.

[46] X. Wang, "Matching records in Multiple Databases Using a Hybridization of Several Technologies," Department of Industrial Engineering, University of Louisville, 2008.

[47] J. Zhou, X. Lv, Y. Mu, X. Wang, J. Li, X. Zhang, J. Wu, Y. Bao and W. Jia, "The accuracy and efficacy of real-time continuous glucose monitoring sensor in Chinese diabetes patients: a multicenter study," vol. 14, no. 8, 2012.

[48] "Wikipedia, the free encyclopedia," [Online]. [Accessed 2016].

[49] W. Su, J. Wang and F. H. Lochovsky, "Record Matching over Query Results from Multiple Web Databases," vol. 22, no. 4, 2010.

[50] F. G. Alizamini, M. M. Pedram, M. Alishahi and K. Badie, "Data quality improvement using fuzzy association rules," 2010.

[51] V. Chanana and A. Koronios, "Data Quality Through Business Rules," 2007.

[52] F. Chiang and R. J. Miller, "Discovering data quality rules," vol. 1, no. 1, 2008.

[53] H. Ruan, D. Yu and Y. Cao, "Research and Implementation of the Platform for Analyzing Data Quality," 2009.

[54] M. S. Shahriar and S. Anam, "Quality Data for Data Mining and Data Mining for Quality Data: A Constraint Based Approach in XML," 2008.

[55] O.H. Choi, J.-E. Lim, H.-S. Na and D.-K. Baik, "An Efficient Method of Data Quality using Quality Evaluation Ontology," 2008.

[56] J. Hipp, U. Güntzer and U. Grimmer, "Data Quality Mining-Making a Virute of Necessity.," in *DMKD*, 2001.

[57] P. Z. Yeh and C. A. Puri, "An Efficient and Robust Approach for Discovering Data Quality Rules," 2010.

[58] E. Davis, "Guide to Expressing Facts in a First-Order Language," 2015.

[59] "Wikipedia, the free encyclopedia," [Online]. Available: https://en.wikipedia.org/wiki/Semantic_heterogeneity..

[60] T. Heath and C. Bizer, Linked data: evolving the web into a global data space, Morgan & Claypool, 2011.

[61] C. Peter, Data matching: concepts and techniques for record linkage, entity resolution, and duplicate detection, Springer, 2012.

[62] T. Devogele, "A New Merging Process for Data Integration Based on the Discrete Fréchet Distance," in *Advances in Spatial Data Handling*, Springer Berlin Heidelberg, 2002.

[63] I. Ullah, "DATA QUALITY MANAGEMENT USING DATA MINING TECHNIQUES," *Islamic Countries Society of Statistical Sciences,* p. 435.

[64] E. Emerson, "Temporal and modal logic," *Handbook of Theoretical Computer Science, Elsevier*, vol. B, 1990.

[65] J. Wijsen, "Certain Conjunctive Query Answering in First-Order Logic," *ACM Transactions on Database Systems,* vol. 37, 2012.

[66] "WIKI," 19 4 2016. [Online]. Available: https://en.wikipedia.org/wiki/Conjunctive_query. [Accessed 11 10 2016].

[67] "Laerd Statistics," 2013. [Online]. Available: https://statistics.laerd.com/statistical-guides/pearson-correlation-coefficient-statistical-guide.php. [Accessed 1 5 2017].

[68] Andale, " Statistics How To Theme by: Theme Horse Powered by: WordPress," 11 2 2013. [Online]. Available: http://www.statisticshowto.com/what-is-the-pearson-correlation-coefficient/. [Accessed 1 5 2017].

CURRICULUM VITAE

Amani Shatnawi

Computer Science

Utah State University

Old Main 404, Utah State University, Logan, UT 84322
E-mail: shatnawi@aggiemail.usu.edu
Phone: 435- 363- 5782

Education

Ph.D. Degree, Computer Science — May,2017
Utah State University, Logan, UT
Thesis Topic: Quality of integrated Person Data
Adviser: Assoc. Professor Stephen Clyde

GPA: 4

M.S. Degree, Computer Science — May, 2010
Jordan University of Science and Technology, Irbid, Jordan
Thesis Topic: Software Prototyping from UML Diagrams Using Language – Independent User Interfaces
Adviser: Assoc. Professor Raed Shatnawi

B.S. Degree Computer Science — June, 2006
Jordan University of Science and Technology, Irbid, Jordan
Graduate Project: E-learning System
Adviser: Assoc. Professor Basel Mahafza

Research Interests

Object-Oriented Analysis and Design, Object-Oriented Modeling, Data Mining, Data Quality, Temporal XML Data, Search Engines.

Experiences

Teaching Assistant / Classes Taught

Developing Dynamic, Database-Driven, Web Applications - CS 2610 — Aug 27, 2014- May 2017

- PhD Student / Teaching Assistant
- Utah State University , Logan, UT , USA
- Course Description: Develop secure, dynamic, database-driven web applications. Learn the fundamentals of building web pages. Add interactive capabilities with client-side and server-side technologies. Learn about information representation, storage, retrieval and transmission in Internet-based environments.
- Reference : Linda Duhadway (linda.duhadway@usu.edu)

C++ Programming Language Sep, 2010 – Dec, 2010

- Part Time Lecturer
- Jordan University of Science and Technology, Irbid, Jordan
- Course Description: C++ programming concepts, variables and basic data types, control structures and loops, functions, arrays, structures, classes and objects, constructors and destructors, inheritance, pointers and references to objects, streams and files.

Computer Skills Sep, 2010 – Dec, 2010

- Part Time Lecturer
- Jordan University of Science and Technology, Irbid, Jordan
- Course Description: This course provides the very basic computer skills to students who have failed in demonstrating such skills in their college admittance test. The course covers topics such as computer components, computer functions and benefits, computer viruses and measure of protection. Also, introduction to operating systems, application software (including word processing, spreadsheets and presentation applications), Internet, e-mail systems, e-learning systems, e-library systems.

Research Assistant

Sept. 2011- May 2014

Software Engineering - Quality of integrated Health Data

- Ph.D Student / Research Assistant
- Utah State University , Logan, UT , USA
- Supervisor: Dr.Stephen Clyde.

Current Research

Temporal XML Search

- Keyword search is used to search the current state of the documents in a collection, but when the documents change as they are edited and deleted users would like to also search the history of the edits and past versions of the documents. This paper extends existing keyword search techniques to support search within a time-varying or temporal document collection. We present a system called Temporal XML Keyword Search (TXKS) that supports temporal search.
- TXKS lets users control which temporal slice, or part of the history, can be searched, and provides two basic search modes: sequenced and nonsequenced. Sequenced search searches within a slice using only the documents that existed during the slice, whereas nonsequenced search explores different slices.
- We provide a temporal data model, show how the model supports slicing, sequenced and nonsequenced search, and how temporal search can be efficiently implemented. Our extensions are (largely) orthogonal to any particular search technique, so this research provides a blueprint for making any search technique temporal.

Master Thesis Research

Software prototyping from UML diagrams using language – Independent User Interfaces

- Ms. Thesis.
- Computer Science Department. Jordan University of Science and Technology. Irbid, Jordan.
- Jan, 2009 – May, 2010.

- Description: I suggest a new approach to generate UI prototype. It provides a six activities process for deriving a UI prototype from the UML diagrams that are enriched with UI information. Based on end user feedback, the UML diagrams and the UI prototype may be iteratively refined. As a result of my approach I build a tool (UI-gen) that automates the generation of UI prototype. This prototype is coded using an XML-based language called the User Interface Markup Language (UIML).
- Outcomes: The tool (UI-gen) will provide a general-purpose presentation of the UI prototype that is implementation-independent from operating systems and platforms. The process of generating these UIs should help developers in producing a rapid prototype that can help developers in their negotiation with customers.

A New Perfect Hashing and Pruning Algorithm for Mining Association Rule

- Advance Data Mining Course Project and Research.
- Computer Science Department. Jordan University of Science and Technology. Irbid, Jordan.
- Hassan Najadat, Assoc. Professor.
- Information System Department. Jordan University of Science and Technology. Irbid, Jordan.
- Sep, 2007 – March, 2008.
- Role: gathering the last researches in this topic, find the new idea, implementing the new algorithm with its own data structure and writing the paper.
- Description: This research presents a new hashing algorithm in discovering association rules among large data of itemsets. Our approach scans the database once and utilizes an enhanced version of priori algorithm like Direct Hashing and Pruning algorithm (DHP). The algorithm computes the frequency of each k itemsets and discovers a set of rules from these frequent k itemsets. Once the expert in the application domain provides the minimum support and a specific size of database is scanned, the pruning phase is utilized to minimize the number of k itemsets generated.

Outcomes: The analysis shows that the new algorithm does not suffer from the collisions, which lead to high accuracy.

Publications

Journal articles

1. Amany Shatnawi and Raed Shatnawi. Generating a Language-Independent Graphical User Interfaces from UML Models, IAJIT Journal

Conferences/Symposium Articles

2. Hassan Najadat, Amani Shatnawi and Ghadeer Obiedat.A New Perfect Hashing and Pruning Algorithm for Mining Association Rule. Volume 2011 (2011), Article ID 652178, Communications of the IBIMA, 8 pagesDOI:10.5171/2011.652178.
3. CE Dyreson, VA Rani, A Shatnawi. Unifying Sequenced and Non-sequenced Semantics. Temporal Representation and Reasoning (TIME), 2015

Honors, awards and fellowships

- A scholarship to pursue Ph.D. in computer science (Sep, 2011 – May, 2016) from Utah State University, Logan, UT.
- A scholarship to pursue M.Sc. in computer science (Sep, 2007 – May, 2010) from King Abdullah II Fund for Development.
- A scholarship to pursue B.S. in computer science (Sep, 2002 – May, 2006) from General Command of the Armed Forces.

Professional training/development

Developing E-learning Systems.

- Computer and Information Center, Jordan University of Science and Technology. Irbid, Jordan.
- Jan, 2006 – Jun, 2006.
- Role:
    - Perform task analysis: Determine the tasks to be taught, identify subtasks and other elements involved, and identify the knowledge, skills, and attitudes required to complete the tasks efficiently and effectively.
    - Perform training needs analysis: Identify the target audience for the training. Identify the shortfall in knowledge, skills, and attitudes of this audience and determine what the target learners need to know.
    - Review existing capabilities: Review existing methods and infrastructure for providing training or meeting learning needs.
    - Implementing part of the system.

Languages

- Arabic native
- English Reading/Writing is very good

Skills

- Programming Languages*:* C++, C# (2003-2012).
- Web Programming: HTML, XHTML, XML,UIML, Java Scripts
- Database Management System: Advanced Access 2000, 2002, Oracle10g, MySQL 5.2 CE, SQL server, Sybase, Oracle (SQL, PL\SQL, and FORMS).
- Web Page Application: ASP.NET.
- Object Oriented Methodologies (Modeling Languages): UML.
- Modeling Tools: Rational Rose, Visual Paradigm.

Related experience

Teaching Assistant	Computer Science Department	Utah State University. Logan, UT.	Aug, 2014 –May 2017
Graduate Research Assistance.	Computer Science Department	Utah State University. Logan, UT.	Oct, 2011 – May,2014
Teaching Assistant workshop	Computer Science Department	Utah State University. Logan, UT.	June,2013
Part-time Lecturer.	Computer Science Department, Information System Department	Jordan University of Science and Technology. Irbid, Jordan.	Sep,2010 – Dec,2010

Teaching Assistance, Lab Supervisor.	Computer Science Department	Jordan University of Science and Technology. Irbid, Jordan.	Sep,2006 – Sep,2007

Professional affiliations/service
None

References

Stephen Clyde	Associate Professor	Utah State University. Logan, UT. Faculty of Engineering. Computer Science Department.	stephen.clyde@usu.edu Fax# (435) 797-3265 Office# (435) 797-2307
Raed Shatnawi	Associate Professor	Jordan University of Science and Technology. Irbid, Jordan. Faculty of Computer and Information Technology. Software Engineering Department.	raedamin@just.edu.jo Phone: (+962)2-7201000 Ext: 23379
Curtis Dyreson	Assistant Professor	Utah State University. Logan, UT. Faculty of Engineering. Computer Science Department.	Curtis.Dyreson@usu.edu Fax# (435) 797-3265 Office# (435) 797-0742